VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Aplikace metod pro redukci dimenzionality

Application methods for dimensionality reduction

2016 Bc. Jakub Orava

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Jakub Orava**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Aplikace metod pro redukci dimenzionality
Application Methods for Dimensionality Reduction

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem této práce je prostudovat metody pro redukci dimenzionality a použít je v různých oblastech. Součástí práce je implementace vybraných metod.

1. Prostudujte teorii k problematice redukce dimenzionality.
2. Vytvořte přehled metod vhodných pro redukci dimenzionality v různých oblastech.
3. Implementujte několik vybraných metod.
4. Vyberte a popište různě rozsáhlé datové kolekce, nad kterými aplikujete vybrané metody.
5. Porovnejte a analyzujte získané výsledky a vhodně je reprezentujte.

Seznam doporučené odborné literatury:

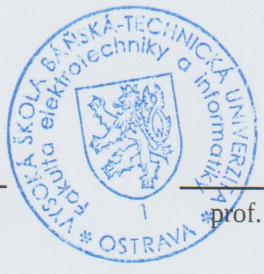
[1] Bartl, E., Rezankova, H., & Sobisek, L. (2011, October). Comparison of Classical Dimensionality Reduction Methods with Novel Approach Based on Formal Concept Analysis. In RSKT (pp. 26-35).
[2] Van Der Maaten, L.J.P., Postma, E.O. & Van Den Herik, H.J. Dimensionality Reduction: A Comparative Review. Journal of Machine Learning Research 10, 1-41 (2009).
[3] Rajaraman, A., & Ullman, J. D. (2012). Mining of massive datasets (Vol. 77). Cambridge: Cambridge University Press.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Mgr. Pavla Dráždilová, Ph.D.**

Datum zadání: 01.09.2015

Datum odevzdání: 29.04.2016



doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 21. dubna 2016 .....

Rád bych na tomto místě poděkoval vedoucí práce, Mgr. Pavle Dráždilové, Ph.D., za všechnu pomoc, ochotu a cenné rady při vypracování této diplomové práce.

Abstrakt

Tato diplomová práce se zabývá možnými aplikacemi vybraných metod pro redukci dimenzionality. Především se zaměřuje na metody Singular Value Decomposition, Principal Component Analysis, Non-negative Matrix Factorization, Kernel Principal Component Analysis a CUR. Tyto metody nacházejí uplatnění v různých oborech. V této práci jsou popsány základní pojmy potřebné pro pochopení problematiky. Dále jsou zde uvedena možná použití a popis vybraných metod. Další část je věnována experimentům, které jsou převážně zaměřeny na kompresi obrázků, ale je zde i popsáno několik použití vybraných metod nad dokumenty. Cílem této práce je seznámit čtenáře s metodami redukce dimenzionality, implementace těchto metod a provedení experimentů nad různými datovými kolekcemi.

Klíčová slova: redukce dimenzionality, Singular Value Decomposition, Principal Component Analysis, Non-negative Matrix Factorization, Kernel Principal Component Analysis, CUR, komprese obrázku, Latent Semantic Indexing, vizualizace dokumentů, detekce témat

Abstract

The master thesis deals with possible applications of selected methods for dimensionality reduction. It focuses mainly on the following methods: Singular Value Decomposition, Principal Component Analysis, Non-negative Matrix Factorization, Kernel Principal Component Analysis and CUR. These methods are used in various areas. In the thesis, basic concepts necessary for understanding of the subject matter are explained. The work continues with possible applications of selected methods and their explanation. Following part deals with experiments, which are mainly focused on an image compression, but several applications of selected methods on documents are also presented there. The aim of the thesis is to acquaint readers with the methods of dimensionality reduction, an implementation of these methods and the performance of experiments with various datasets.

Key Words: dimensionality reduction, Singular Value Decomposition, Principal Component Analysis, Non-negative Matrix Factorization, Kernel Principal Component Analysis, CUR, image compression, Latent Semantic Indexing, document visualization, topic detection

Obsah

Seznam použitých zkratek a symbolů

CD	–	Compact Disc
DNA	–	Deoxyribonucleic acid
FA	–	Factor analysis
GUI	–	Graphical User Interface
ICA	–	Independent Component Analysis
KPCA	–	Kernel Principal Component Analysis
LDA	–	Linear Discriminant Analysis
LEM	–	Laplacian Eigenmaps
LLE	–	Locally Liner Embedding
LSI	–	Latent Semantic Indexing
MDS	–	Multidimensional Scalling
mRNA	–	Messenger ribonucleic acid
NMF	–	Non-negative Matrix Factorization
PCA	–	Principal Component Analysis
RGB	–	Red-Green-Blue
SVD	–	Singular Value Decomposition
TF-IDF	–	Term Frequency - Inverse Document Frequency

Seznam obrázků

Seznam tabulek

1 Úvod

Velká data neboli big data jsou v dnešní době velmi probíraným tématem. Zásluhu na tom má tzv. čtvrtá průmyslová revoluce a s ní související pojem internet věcí. V internetu věcí je sbíráno a analyzováno velké množství dat, které je vhodné si před prováděním výpočtů a analyzováním upravit. Úprava je zde myšlena ve smyslu, že se redukuje dimenze problému. Pro redukci dimenzionality existuje velké množství metod.

Téma diplomové práce je aplikace metod pro redukci dimenzionality nad různými datovými kolekcemi. Díky metodám redukce dimenzionality je možné snížit dimenzi problému, a tím ušetřit spoustu výpočetního času, neboť většina těchto metod rozkládá problém reprezentovaný maticí na několik menších matic. K analýze jedné velké matice by bylo třeba daleko více času, než například u analýzy tří menších matic. Dalším důvodem použití redukce dimenzionality je, že sesbíraná data (obrázky, zvukové signály, DNA) jsou obvykle vícedimenzionální a pro jejich lepší pochopení je třeba dimenzi problému redukovat na dvě nebo tři dimenze. Díky redukci je reprezentace nasbíraných dat smysluplnější [1].

Druhá kapitola popisuje matematické pojmy, s kterými se může čtenář v průběhu čtení práce setkat. Pojmy spadají do oblasti lineární algebry a statistiky.

Kapitola třetí poskytuje informace o redukci dimenzionality a rozdělení jejich metod. Jsou zde uvedeny i metody, které nebyly implementovány v rámci diplomové práce, ale patří do této problematiky.

Čtvrtá kapitola prezentuje přehled oborů, kde lze metody redukce dimenzionality aplikovat. Oborů je poměrně mnoho - zdravotnictví, chemický průmysl, dopravní sítě a mnoho dalších.

Popis jednotlivých metod, které byly vybrány pro tuto diplomovou práci, poskytuje kapitola pátá. Metody jsou zde popsány jak prostým textem, tak i algoritmicky.

V šesté kapitole se čtenář seznámí s detaily implementace. Mimo popis jednotlivých komponent jsou zde popsány i použité datové kolekce a jejich reprezentace pomocí matic. Těmito datovými kolekcemi jsou obrázky a dokumenty. Pro jednoduché ovládání byly vytvořeny dvě desktopové aplikace s grafickým uživatelským rozhraním. Kapitola také poskytuje přehled o použitých knihovnách třetích stran.

Další tři kapitoly se již zabývají experimenty nad zvolenými datovými kolekcemi. Kapitola sedmá popisuje výsledky experimentů nad obrázky. Osmá kapitola demonstruje na jednoduchých příkladech použití vybraných metod nad kolekcemi dokumentů. V kapitole deváté jsou prezentovány výsledky detekce témat a reprezentace dokumentů nad větší kolekcí dat. Z důvodu velkého množství výsledků z experimentů nad obrázky, je většina z nich uvedena v přílohách.

2 Matematické základy

2.1 Řídká matice

Řídká matice obsahuje velké množství nulových prvků. Většinou jsou tyto matice kvůli velkému množství nulových prvků také velmi rozměrné, tzn. tisíce až statisíce řádků a sloupců. Řídká matice však není matematický pojem a nelze ji přímo definovat. Jednou z „definic“ může být tato: *Matici považujeme za řídkou, jestliže má takový počet nulových prvků, že se vyplatí jich využít (ve smyslu efektivity práce)* [2]. Často se řídkou matici reprezentují rozsáhlé grafy, kde počet nenulových prvků je přibližně $n \cdot k$, kde $k \ll n$.

2.2 Ortogonální matice

Ortogonální matice je čtvercová matice A, obvykle reálná, pro kterou platí $A^T A = E$, kde A^T je transponovaná matice k A a E je jednotková matice [3].

2.3 Pseudoinverzní matice

Pseudoinverzi je třeba použít v případě, kdy se vytváří matice inverzní, avšak tato matice nemůže být klasickou inverzí invertována. Mezi tyto matice patří matice obdélníková a singulární. Většinou se v literatuře uvádí pojem Moore-Penroseova pseudoinverze. Pseudoinverze matice A se označuje jako A^+. Aby se jednalo o Moore-Penroseovu pseudoinverzi, musí splňovat následující podmínky:

1. $A \cdot A^+ \cdot A = A$
2. $A^+ \cdot A \cdot A^+ = A^+$
3. $(A \cdot A^+)^T = A \cdot A^+$
4. $(A^+ \cdot A)^T = A^+ \cdot A$

Pokud pseudoinverzní matice A^+ splňuje tyto podmínky, je jedinečná. Pro každou matici tak existuje právě jedna Moore-Penroseova pseudoinverzní matice.

Pseudoinverzní matici lze vypočítat například pomocí rozkladu na součin matic, iterační metodou Ben-Israela a Cohena nebo pomocí metody používající singulární rozklad [4].

2.4 Rozptyl

Rozptyl [5] je jednou z charakteristik variability a vyjadřuje, jak moc se liší jednotlivé hodnoty proměnných od průměru celého souboru. Rozlišují se dva rozptyly - základní a výběrový. Rozdíl je v přesnosti výpočtů jednotlivých rozptylů s tím, že výběrový rozptyl je přesnější. Rovnice 1

popisuje výběrový rozptyl.

$$D(X) = \frac{\sum_{i=1}^{n}(X_i - X)^2}{(n-1)} \tag{1}$$

2.5 Kovariance a korelace

2.5.1 Kovariance

Kovariance je statistická míra k určení lineární závislosti dvou veličin, tedy jejich vzájemného vztahu. Kovariance je střední hodnota součinu a odchylek obou náhodných veličin X a Y od jejich středních hodnot.

Definice:
Uvažujme náhodný vektor $Z = \binom{X}{Y}$. Kovariancí náhodných veličin X a Y rozumíme hodnotu $cov(X,Y) = E[(X - E(X))(Y - E(Y))]$ [6].

Náhodné veličiny mohou být závislé pozitivně nebo negativně. Při pozitivní závislosti, kdy je kovariance větší než nula platí, že obě hodnoty rostou společně. V případě negativní závislosti ($cov(X,Y) < 0$) platí, že jedna hodnota roste a druhá klesá.

Vzhledem k tomu, že kovariance určuje pouze pozitivní či negativní závislost daných veličin, ne sílu závislosti, je potřeba znát korelaci.

2.5.2 Korelace

Korelace určuje, jak silně jsou dvě náhodné veličiny závislé. Korelaci značíme ρXY.

Definice:
Uvažujme náhodný vektor $Z = \binom{X}{Y}$. Korelací [7] náhodných veličin X a Y rozumíme hodnotu:

$$\rho XY = \frac{cov(X,Y)}{\sqrt{cov(X,X)cov(Y,Y)}} = \frac{cov(X,Y)}{\sqrt{D(X)D(Y)}} \tag{2}$$

2.6 Kovarianční a korelační matice

2.6.1 Kovarianční matice

Vzhledem k tomu, že v metodách redukcí dimenzionality se prakticky nelze setkat s takovými daty, že by se porovnávaly pouze dvě náhodné veličiny, ale mnohem více náhodných veličin, je potřeba nadefinovat další pojem, tím je kovarianční matice. Bylo by možné pro každou dvojici náhodných veličin spočítat kovarianci. Při n náhodných veličinách by bylo potřeba spočítat

*n!/(n-2)!*2* kovariančních hodnot, tedy při kolekci 2000 dokumentů by těchto kovariančních hodnot bylo téměř osm miliónů.

Kovarianční matice má tvar takový, že na diagonále se nacházejí rozptyly náhodných veličin a kolem ní se vyskytují jednotlivé kovarianční hodnoty. Rozměry matice jsou $n \times n$, kde n je počet dimenzí [8].

$$M = \begin{pmatrix} cov(X,X) & cov(X,Y) & cov(X,Z) \\ cov(Y,X) & cov(Y,Y) & cov(Y,Z) \\ cov(Z,X) & cov(Z,Y) & cov(Z,Z) \end{pmatrix}$$

Kovarianční matice je symetrická okolo diagonály, protože $cov(X,Y) = cov(Y,X)$.

2.6.2 Korelační matice

Korelační matice je podobná kovarianční matici s tím, že prvky matice nejsou tvořeny kovariancemi dvojic náhodných veličin, ale jejich korelacemi. Na diagonále budou tedy samé jedničky, protože platí $D(X) = \sqrt{D(X)^2}$.

2.7 Vlastní vektory a vlastní čísla

Nechť $A \in \mathcal{L}(\mathcal{V})$ je lineární transformace definovaná na vektorovém prostoru $\mathcal{V}$. Jestliže existuje nenulový vektor $u \in \mathcal{V}$ a skalár λ tak, že

$$Au = \lambda u, \tag{3}$$

pak se λ nazývá *vlastní číslo* transformace A, u se nazývá *vlastní vektor* příslušný k λ a (λ, u) se nazývá vlastní dvojice transformace A [9].

2.8 Singulární čísla

Nechť $A \in \mathbb{R}^{m,n}$. Odmocniny nenulových vlastních čísel matice $A^T A$ nazýváme singulárními čísly matice A [9].

2.9 Hlavní komponenty

Hlavní komponenty jsou nové proměnné, které zachovávají rozptyl, ale eliminují kovariance [10]. Hlavní komponenty odpovídají vlastním vektorům matice kovariance či korelace u metody PCA.

Vlastnosti:

1. Hlavní komponenty pc_i jsou vzájemně ortogonální.
2. Rozptyly pc_i jsou maximální pro $i = 1$ a postupně klesají.
3. Hlavní komponenty nulových λ_i jsou téměř konstantní.

3 Redukce dimenzionality

Mnoho dat v dnešní době může být reprezentováno jako rozsáhlá matice. Například to mohou být sociální či dopravní sítě nebo web. A proto je vhodné využít možností redukce dimenze. Redukce dimenzí někdy mapuje data do nižších dimenzí tak, aby rozptyl těchto dat byl nulový. Další možností mapování je nalezení vhodného podprostoru pro vstupní data, který je obvykle menší než jejich výchozí prostor s tím, že informativní hodnota původních dat bude co nejvíce zachována. Metody lze použít jak pro kompresi dat, abychom usnadnili chod algoritmů a ostatních metod, pro získání skryté struktury dat nebo také pro snadnější pochopení vstupních dat.

Některé metody rozkládají vstupní matici na několik menších matic, které jsou pak snadno uchopitelné pro další výpočty. Většina metod počítá s mnohem menším počtem řádků a sloupců z původní matice, příkladem může být metoda NMF.

Klasické rozdělení metod uvádí práce Jana Kaliny a Jurjena Duintjera Tebbense [11]

1. Selekce rysů (proměnných či příznaků)
2. Extrakce rysů (proměnných či příznaků)
   a. Lineární
      i. Principal Component Analysis (PCA)
      ii. Singular Value Decomposition (SVD)
      iii. Non-negative Matrix Factorization (NMF)
      iv. Linear Discriminant Analysis (LDA)
      v. Canonical Correlations Analysis (CCA)
      vi. Sufficient Dimensionality Reduction (SDR)
      vii. Independent Component Analysis (ICA)
      viii. Factor Analysis (FA)
   b. Nelineární
      i. Multidimensional Scaling (MDS)
      ii. Kernel Principal Component Analysis (KPCA)
      iii. Locally Linear Embedding (LLE)
      iv. Laplacian Eigenmaps (LEM)
      v. Semidefinite Embedding (SDE)
      vi. Isomapy
      vii. t-Distributed Stochastic Neighbor Embedding (t-SNE)
      viii. Sammonova projekce

Toto však není jediné rozdělení metod, dle práce Sorzana, Vargase a Pascual-Montany lze metody používající extrakci rysů rozdělit na [12]:

1. Metody založené na statistice a teorii informací - PCA, Generative Topographic Mapping (GTM), KPCA, MDS, FA, ICA
2. Metody založené na slovnících - NMF, Principal Tensor Analysis (PTA), Generalized SVD (GSVD)
3. Metody založené na projekci
   a. Projekce do zajímavých směrů
   b. Projekce do větších dimenzí - KPCA, LEM, LLE

Další rozdělení, které uvádí ve své práci Saul a kolektiv, vypadá následovně [13]:

1. Lineární metody - PCA
2. Grafově založené metody - Isomapy, LLE, LEM
3. Kernel metody - KPCA

Tato práce se řídí klasického rozdělení metod na lineární a nelineární. Dle tohoto rozdělení byly vybrány i metody použité a naimplementované v této práci. Mezi vybrané lineární metody se řadí SVD, PCA a NMF. Aby zde však měly zastoupení i metody nelineární, byla zvolena metoda KPCA. Navíc je v této práci začleněna metoda CUR pro rozklad matic, která není v literatuře přímo řazena ke skupině lineárních či nelineárních metod. Tyto metody budou popsány podrobněji v kapitole 5.

3.1 Selekce rysů

Selekce rysů je výběr takové podmnožiny proměnných, které budou pro analýzu podstatné. Proměnné, které nejsou podstatné, mohou být nějaké redundantní nebo mají malou informativní hodnotu pro analýzu.

3.2 Extrakce rysů

Extrakce rysů naopak používá pro výpočet všechny proměnné, které poté převádí na určité kombinace tím, že data z $\mathbb{R}^n$, převede do prostoru $\mathbb{R}^k$, kde $k < n$. To znamená, že vytváří transformaci z n-dimenzionálního prostoru do k-dimenzionálního prostoru. Tato transformace může být lineární či nelineární.

3.3 Popis dalších metod redukce dimenzionality

3.3.1 Lineární diskriminační analýza (LDA)

Lineární diskriminační analýza provádí redukci dimenzí tím, že promítne vstupní data do lineárního podprostoru, který se skládá z vektorů maximalizujících vzdálenost mezi třídami [14].

Obecný přístup LDA je podobný metodě PCA, avšak namísto hledání komponent, které poskytují největší rozptyl vstupních dat, hledá LDA osy, které maximalizují oddělení mezi více třídami.

Používá se jako krok pro předzpracování dat při klasifikaci vzorků a při použití v aplikacích strojového učení.

Postup metody je velmi podobný metodě PCA [15]:

1. Spočítej průměr ($\overline{x}_i$) pro každou třídu ve vstupních datech.
2. Spočítej S-matici, kde

$$S = \sum_{j=1}^{n} (\mathbf{x}_j - \overline{\mathbf{x}})_i (\mathbf{x}_j - \overline{\mathbf{x}})_i^T, \forall i = 1, ..., m$$

3. Spočítej vlastní vektory a jim odpovídající vlastní čísla ze získané S-matice.
4. Seřaď vlastní vektory podle vlastních čísel a vyber k prvních vlastních vektorů.
5. Udělej projekci vzorků do nového podprostoru pomocí vybraných vlastních vektorů.

Mezi PCA a LDA je tedy rozdíl v algoritmu ten, že nepoužívá matici kovariance/korelace ale S-matici.

3.3.2 Multidimenzionální škálování (MDS)

Multidimenzionální škálování je přístup k nalezení skrytých atributů či dimenzí, které ovlivňují, jak subjekty vyhodnocují danou sadu objektů nebo podnětů. Výsledné dimenze by měly analytikovi umožnit pochopit odlišnosti či podobnosti mezi pozorovanými objekty. Vstupem pro MDS může být libovolná matice podobnosti či nepodobnosti. MDS zobrazuje vstupní data pomocí jejich vzdáleností jako graf či geometrický útvar.

MDS nachází uplatnění v mnoha oborech, jako jsou sociologie, psychologie, fyzika, politické vědy či biologii [16].

Typů MDS je několik – klasické, metrické (kvalitativní) a nemetrické (kvantitativní).

Většinou je výstup reprezentován dvourozměrným či trojrozměrným euklidovským prostorem, může však být reprezentován i prostorem, který není euklidovský a má větší počet dimenzí než tři. V tomto prostoru můžeme vidět, které objekty jsou si blízké a které naopak. Například

vzdálenosti mezi městy. Je zřejmé, že v prostoru si budou velmi blízká města, jako jsou Budapešť, Bratislava a Vídeň, avšak Londýn a Dublin si budou blízké, ale od první skupiny budou velmi vzdáleny. MDS tedy poskytuje prostorovou představu o vzdálenostech objektů.

Z technického hlediska se lze na princip MDS dívat tak, že hledá množinu vektorů v k-dimenzionálním prostoru. Výběr vektorů se liší dle vybraného typu metody. U klasického MDS se vybírají vektory na základě vlastních čísel a u metrického dle ztrátové funkce, která je vypočtena mezi původními a transformovanými daty.

Postup algoritmu MDS je následující:

1. Sestav matici blízkostí.
2. Aplikuj dvojité centrování, kde výsledkem bude matice B.
3. Vyber m největších vlastních čísel z matice B a k nim m korespondujících vlastních vektorů.
4. Získej a vrať m-dimenzinální prostorové uspořádání n objektů.

Pro lepší pochopení metody MDS lze doporučit práci Floriana Wickelmaiera „An introduction to MDS“ [17].

3.3.3 Isomapy

Isomapy usilují o zachování vnitřní geometrie dat, zachycených v geodeticky rozličných vzdálenostech mezi všemi dvojicemi jednotlivých bodů. Pro sousední body poskytuje dobrou aproximaci ke geodetické vzdálenosti jejich vzdálenost ve vstupním prostoru. Pro body, které jsou vzdálené, může být geodetická vzdálenost aproximována několika malými skoky mezi sousedními body. Aproximace jsou počítány efektivně díky tomu, že algoritmus hledá nejkratší cestu v grafu mezi jednotlivými uzly pomocí hran spojujících jednotlivé sousední body.

Mezi Isomapami a MDS existuje souvislost, kterou lze pochopit tak, že MDS pracuje s maticemi, které vyjadřují vzdálenosti mezi každým párem bodů v datech. Otázkou je, jak tyto vzdálenosti získat. Jednou z možností by mohl být výpočet euklidovských vzdáleností mezi každým párem, anebo použít právě Isomap, které využívají algoritmu k-nejbližších sousedů.

Isomapy umožňují nalézt nelineární stupně volnosti, které pochází z dat získaných složitým fyzickým pozorováním, jako jsou fotky obličeje z různých úhlů a při různých světelných podmínkách nebo rukopis člověka.

Tento přístup k redukci dimenzionality kombinuje nejlepší vlastnosti dvou algoritmů – PCA a MDS. Mezi ty nejlepší vlastnosti patří efektivita a záruka asymptotické konvergence [18].

Postup výpočtu pro Isomapy [19]:

1. Urči sousedy (algoritmus k-nejbližších sousedů) a sestroj graf těchto sousedů.
2. Spočítej nejkratší cestu mezi dvěma uzly (Floydův nebo Dijsktrův algoritmus).
3. Proveď MDS.

4 Použití vybraných metod

4.1 SVD

Singular Value Decomposition je metoda rozkladu matice, která rozkládá matici A o rozměrech $m \times n$ na tři dílčí matice - U, Σ a V. Každá ze tří matic má určitý rozměr založený na původní matici a matici Σ. Jak jsou rozměry jednotlivých matic dány, bude popsáno v kapitole 5. U je komplexní unitární matice levých singulárních vektorů. Matice Σ je diagonální matice, která na své diagonále neobsahuje záporná čísla a okolí diagonály je vyplněno nulami. Tato nezáporná čísla se nazývají singulární čísla. Matice V je transponovaná reálná či komplexní unitární matice pravých singulárních vektorů.

SVD má mnoho použití. Jedno z možných užití metody SVD je odstranění šumu tak, že se SVD aplikuje na získaná data a snaží se odstranit složku způsobující šum. Také určuje, jak moc „zašumněná“ získaná data jsou [20].

Metodu SVD lze použít také na kompresi obrázků díky tomu, že lze obrázky reprezentovat jako matici. Matice obrázku se rozloží pomocí SVD a následným užitím nízkoúrovňové (low-rank) aproximace se určí, kolik singulárních čísel ponechat. V mnoha případech stačí i polovina počtu sloupců matice ke kompresi takové, že původní obrázek není téměř okem rozeznatelný od původního. Komprimované obrázky lze také využívat k rozeznávání obličejů, např. v kriminalistickém softwaru [21].

Taktéž metoda LSI (Latent Semantic Index) používá SVD. LSI je metoda pro vyhledávání ve velkém množství dokumentů pomocí klíčových slov a SVD využívá k zjišťování podobnosti jednotlivých dokumentů. Dokumenty a jejich klíčová slova jsou reprezentovány jako matice o m řádcích a n sloupcích, kde v řádcích jsou slova a ve sloupcích počet jejich výskytů v jednotlivých dokumentech. [22].

SVD nachází i uplatnění v biomedicíně. Například v třídění a porovnávání dat mRNA získaných z tzv. mikropolí. Mikropole si lze představit jako snímek obsahující velké množství skvrn. Cílem je nalezení toho, zda vnější faktory souvisí z těmi vnitřními. Vnější faktory lze vnímat jako nemoci a vnitřní jako zvýšenou hladinu určitého genu oproti hladinám ostatních genů [20].

SVD má mnoho aplikací a byly vyjmenovány pouze některé z nich. SVD může být použito i ve statistice k analýze rozptylu, v obchodním prostředí např. pro analýzu akcií několika firem v určitém časovém období, apod.

4.2 PCA

Principal Component Analysis převádí původní matici X o rozměrech $m \times n$ na matici $Y = W^T.X$, kde W je projekční maticí vybraného počtu vlastních vektorů. Tím jsme nalezli vhodný k-dimenzionální podprostor z n-dimenzionálního prostoru, kde k je počet vybraných vlastních vektorů. Nově vzniklá matice Y bude mít, rozměry $m \times k$, kde $k < n$.

PCA lze využít ke kompresi obrázků. Čím nižší počet hlavních komponent zvolíme, tím bude obrázek méně rozeznatelný. Nejvhodnější poměr počtu zvolených hlavních a původních hlavních komponent pro účinnou kompresi je 5:1. V případě obrazových dat ji lze použít, stejně jako SVD, pro rozpoznávání a identifikaci obličejů [23].

Další doménou aplikace může být i zpracování zvuku. Pokud máme několik zdrojů zvuku například skupinu lidí na oslavě a každý mluví najednou, můžeme odseparovat jednotlivé zdroje zvuku od zdrojů šumění a poslechnout si tak každého zúčastněného zvlášť. Podmínkou však je, aby signály od zdrojů zvuku byly lineární a měly normální rozdělení. Vhodnější je pro tuto aplikaci metoda ICA (Independent Component Analysis) [24].

Výsledek metody PCA je možné také použít pro vizualizaci dokumentů do dvoudimenzionálního nebo třídimenzionálního prostoru.

Metoda PCA může použít SVD jako krok svého algoritmu, a proto lze použít PCA pro podobné aplikace jako SVD. Kromě těch výše zmíněných lze PCA použít i na analýzu obchodních nebo volebních dat. A stejně jako SVD nachází uplatnění i v biomedicíně při analýze mikropolí v mRNA.

4.2.1 Kernel PCA

Kernel PCA se oproti PCA liší v tom, že dimenzi redukuje nikoliv lineárně, ale nelineárně. To se hodí při použití dat, která mají nelineární strukturu, a tak je nelze reprezentovat v lineárním prostoru. Kernel PCA k tomu využívá tzv. kernel funkci, která mapuje původní n-dimenzionální prvky na větší k-dimenzionální prostor prvků, kde $n < k$ tím, že vytvoří nelineární kombinace původních prvků. Metodu kernel PCA lze použít pro stejné aplikace jako PCA. Důležité však je, že kernel PCA je přesnější, což může být velkým benefitem např. při rozpoznávání nikoliv samotného obličeje, ale výrazu (úsměv, výkřik, překvapeni, apod.) [25].

4.3 CUR

CUR nese název po počtu a pojmenování matic, na které rozkládá původní matici A. Matice A má rozměry $m \times n$. Matice C je tvořena vybranými sloupci matice A, tedy matice R je tvořena vybranými řádky matice A. Matice U je dimenze $c \times r$, kde c značí počet vybraných sloupců matice A a r počet vybraných řádků matice A.

CUR je nová metoda pro analýzu dat. Oproti SVD je vhodnější pro řídké matice. CUR lze stejně jako SVD použít ke kompresi obrázků nebo analýze DNA [26].

Metodu CUR je také vhodné použít v případě, kdy chceme získat z malého množství dat množství větší. Například v tzv. doporučovacích systémech, kdy je konkrétní zboží doporučováno určitému zákazníkovi či skupině zákazníků. Problémem je, že nejsou sesbírány data o všech zákaznících a jejich nákupech. Díky použití metody CUR nám postačí i malé množství dat, z kterého metoda CUR vytvoří množství větší a lze tak získat kombinaci uživatelů a výrobků. Díky

těmto kombinacím může například majitel internetového obchodu úspěšně cílit určité produkty na skupinu svých zákazníků a zvýšit tak své tržby [20].

CUR lze využít i při sledování dopravních systémů, kdy je sesbíráno velké množství dat z dopravní sítě. Takovéto velké množství dat lze těžko reprezentovat tak, aby bylo zřejmé, co přesně znamenají. Proto je vhodné data komprimovat. Lze samozřejmě použít například i výše zmíněnou metodu PCA, tato metoda však na rozdíl od CUR vrací špatně čitelné výsledky, kvůli tomu, že hlavní komponenty obsahují velký počet spojů v konkrétní dopravní síti. Oproti tomu model vrácený metodou CUR obsahuje komponenty, které odpovídají jednotlivým spojům v síti. Těchto komponent nemusí být pro stejný účinek komprese použito mnoho. Mimo kompresi již nasbíraných dopravních dat, je také možné CUR použít i pro kompresi při snímání dat [27].

Další uplatnění metody CUR je v chemickém průmyslu, kdy je využívána u hmotnostní spektrometrie. Ta umožňuje prostorové mapování chemického složení, složitých biologických vzorků tím, že vybere ionty a pozice ze snímku pro identifikaci sloučenin. Tato technika je náročná i za použití moderních přístrojů, protože je zde vybíráno velké množství nezpracovaných dat. Proto se používají metody pro redukci dimenzí. Oproti CUR vrací například PCA či NMF lineární kombinace aktuálních dat, které jsou těžko interpretovatelné. CUR poskytuje prokazatelně dobré nízko-úrovňové aproximace, které jsou vlastně skutečnými ionty a pozicemi. Zajišťuje tedy lepší čitelnost výsledků [28].

4.4 NMF

Non-negative matrix factorization rozkládá původní matici A o rozměrech $m \times n$ na dvě matice - W a H. W je matice $m \times r$ a nazývá se maticí faktorů. Matice H neboli směšovací matice má rozměry $r \times n$. V obou případech musí platit, že $r \leq m$.

Jako všechny předešlé metody lze i NMF aplikovat na kompresi obrázků [29]. V obrazovém zpracování se může také využít pro nalezení určitého objektu v pohybu ve video sekvenci. Např. nalezení konkrétního automobilu na dálnici [30].

Další doména využití NMF je získávání dat z dokumentů [31]. První aplikací je shlukování dokumentů tím, že je uspořádá do hierarchie, která následně umožňuje snadné vyhledávání a rychlou orientaci v dokumentech. Další aplikací na textová data je nalézání témat (topiců). Příkladem může být práce o automatické detekci tématu z textu písně [32]. Práce pojednává o tom, že kterákoliv píseň, u které je znám text, může být na základě slov, které text písně obsahuje, zařazena do určité kolekce písní. Tuto kolekci si uživatel poté může procházet a pouštět si písně ne dle žánru, ale tématu (písně o lásce, o počasí apod.). Mezi další aplikace patří analýza e-mailů, sledování trendů.

Se sledováním trendů nebo spíše detekcí souvisí další doména, ve které NMF nachází své uplatnění, a to jsou sociální sítě. První aplikací je detekce trendů a komunit v sociálních sítích (blogy, Twitter, Facebook) [33]. Komunitou může být myšlena například skupina blogů nebo účtů na Twitteru. Komunity spolu komunikují na základě nějaké události. Díky komunitám můžeme analyzovat, jak často spolu dané účty v průběhu času komunikují. Další aplikací může

být, podobně jako u metody CUR, využití NMF v tzv. doporučovacích systémech. Tato aplikace se samozřejmě netýká jen sociálních sítí, ale i internetových obchodů či zpravodajských serverů.

Aplikací NMF je obrovské množství. Stejně jako SVD či PCA nalézá i NMF aplikaci v biomedicíně při analýze „mikropolí“ nebo analýze DNA. Používá se také pro analýzu obchodních dat, apod.

5 Popis vybraných metod

5.1 SVD

SVD vytváří rozklad původní matice A tak, že platí $A = U\Sigma V^T$, kde U a V jsou ortogonální matice a Σ je matice diagonální, která neobsahuje žádné záporné prvky. Takovýto rozklad lze vidět na obrázku 1. Na to, jak spočítat SVD existuje několik algoritmů.

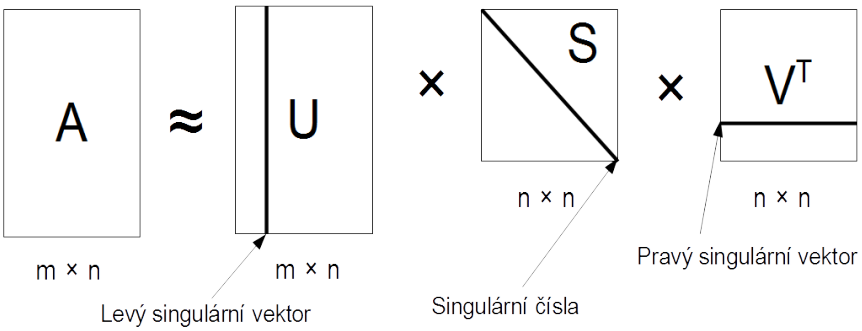
Rozměry matic U a V vyplývají z rozměrů matic A a Σ. Mějme p a q, kde p je počet řádků a q naopak počet sloupců matice Σ. Dále definujme m a n jako počet řádků a sloupců matice A. Matice U bude mít rozměry $m \times p$, kde $p \leq m$ a V bude mít rozměry $n \times q$, kde $q \leq n$.

Existují tři standardní formy SVD [34]:

1. Pokud $p = m$ a $q = n$, poté matice Σ bude mít rozměry $m \times n$, tedy stejné rozměry jako matice A.

2. Pokud $p = q = min\{m, n\}$, tak matice Σ bude čtvercová. Viz. obrázek 1.

3. Dalším případem je tzv. redukované SVD, ve které vystupuje ještě parametr r, který značí, že se bude počítat pouze s prvními r vlastními čísly. Takže platí, že $p = q = r$ a matice Σ je čtvercová.



Obrázek 1: Rozklad matice A pomocí metody SVD

Velmi důležitým pojmem jsou singulární vektory a singulární čísla. S nimi souvisí právě všechny matice z rozkladu. Matice U obsahuje ve sloupcích tzv. levé singulární vektory původní matice A, naopak matice V v řádcích obsahuje tzv. pravé singulární vektory matice A. Co se týče diagonální matice Σ, na její diagonále se nacházejí singulární čísla seřazena ve vzestupném pořadí.

Následuje příklad rozkladu matice o rozměrech 2×3 :

$$A = \begin{pmatrix} 2 & 3 & 6 \\ 1 & 5 & 4 \end{pmatrix}$$

Výsledný rozklad bude vypadat takto:

$$A = \begin{pmatrix} 0.736565 & -0.676367 \\ 0.676367 & 0.736565 \end{pmatrix} \begin{pmatrix} 9.30855 & 0 \\ 0 & 2.08588 \end{pmatrix} \begin{pmatrix} 0.230916 & -0.2954 & -0.927047 \\ 0.600687 & 0.792821 & -0.103005 \\ 0.76541 & -0.53308 & 0.360518 \end{pmatrix}$$

Po pronásobení těchto tří matic dostaneme zpět původní matici.

Problémem redukce dimenzí pomocí metody SVD je najít vhodný k-dimenzionální podprostor z n-dimenzionálního prostoru daného původní maticí. Tento podprostor musí co nejvíce odpovídat původnímu prostoru. Pro vysvětlení pojmu vhodný podprostor je třeba si představit matici $m \times n$ jako n bodů v nějakém m-dimenzionálním prostoru. Nejvhodnější podprostor je takový, který na základě vzdáleností mezi body v původním prostoru a podprostorem minimalizuje sumu čtverců těchto vzdáleností. Například si lze představit přímku jako jednodimenzionální podprostor a nyní je nutné najít nejmenší sumu čtverců takovou, že sumy čtverců vzdáleností bodů prostoru na tuto přímku budou co nejmenší.

Ve zkratce je tedy redukce dimenzí pomocí metody SVD založena na tom, kolik ponecháme singulárních vektorů a čísel.

Jak již bylo zmíněno, pro výpočet SVD existuje několik algoritmů [34]. Mezi ně patří:

1. Golub-Reinsch algoritmus
2. Bidiagonální SVD s vysokou relativní přesností (Demel-Kahan)
3. Bidiagonální singulární hodnoty s vysokou relativní přesností
4. Bidiagonální singulární hodnoty dle půlení
5. SVD pomocí rozděl a panuj (DC_SVD)
6. Biortogonalizační SVD (Jednostranná Jacobiho metoda)
7. Jacobiho rotace

Tento seznam slouží pouze pro představu množství algoritmů řešících singulární rozklad. Vzhledem k tomu, že se práce samotná nezabývá pouze metodou SVD, ale i čtyřmi dalšími, bude popsán pouze jeden z algoritmů. Nejpoužívanějším algoritmem je první v seznamu - Golub-Reinsch [35].

Algoritmus Golub-Reinsch se skládá ze dvou kroků. Prvním krokem je převedení původní matice na matici bidiagonální tím, že aplikuje Householderovy transformace a druhým krokem je opakované použití metody, která využívá ortogonální transformace k vytvoření diagonálních

matic z matice bidiagonální. Algoritmus Golub-Reinsh je nejvíce doporučovaný pro svou účinost a stabilitu. Algoritmus může mít vyšší výpočetní kapacitu potřebnou k vytvoření bidiagonální matice z matice původní. Tento krok se musí totiž provést i tehdy, když je matice jen mírně odchýlena od matice bidiagonální. Algoritmus je sériové povahy, a proto jej není možno používat pro paralelní výpočty. Vhodnějším algoritmem pro paralelní výpočty by mohl být alogritmus Jacobiho rotací.

5.2 PCA

Účelem metody PCA je analyzovat data k identifikaci a vyhledání vzorů. Vzory slouží k určení podobnosti nebo rozdílnosti jednotlivých prvků. Díky vzorům redukuje PCA dimenzi dat s minimální ztrátovostí informací tím, že původní n-dimenzionální prostor zobrazí do menšího, k-dimenzionálního prostoru. To znamená, že snížením redukce dimenze budou data i nadále dobře reprezentovat původní problém.

PCA se snaží najít hlavní komponenty, které by maximalizovaly rozptyl v analyzovaných datech. Obecně se hledají osy s maximálním rozptylem, tedy takové, že v jejich okolí jsou data nejvíce rozptýlena.

Hlavní otázkou, a to nejen pro metodu PCA, ale i jiné metody je „Jak vypadá 'dobrý' podprostor?“. K tomu poslouží právě vlastní vektory a vlastní čísla komponent. Vlastní číslo udává velikost konkrétního vlastního vektoru. Další možností je pozorovat vlastní čísla. Čím vyšší hodnotu vlastní číslo má, tím větší má informativní hodnotu, naopak vlastní čísla s hodnotami nulovými či blízkými nule informativní hodnotu nemají téměř žádnou. Při konstrukci nového podprostoru se využijí vlastní čísla s velkou informativní hodnotou a ta s nízkou do nového podprostoru začleněna nebudou [36].

5.2.1 Volba optimální velikosti podprostoru

Pro určení optimálního podprostoru, čili parametru k, je možné použít tzv. „elbow“ techniku. Elbow technika pomáhá u metod, které pracují s vlastními či singulárními čísly určit optimální počet zachovaných vlastních/singulární čísel. U PCA se volí tak, aby pokryl určité procento variability, např. 99%.

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{j=1}^{n} \lambda_j} \leq ratio \tag{4}$$

Elbow technika na základě sumy k vlastních čísel[1] dělená sumou všech vlastních čísel specifikuje poměr, který určuje, že k vybraných vlastních čísel zachovává určité procento variability, které je v rovnici 4 vyjádřeno pomocí *ratio* [37].

Druhá možná technika pro výběr optimálního počtu vlastních čísel je taková, že se vyberou pouze ta vlastní čísla, jejichž hodnota je větší něž průměr všech vlastních čísel [38]. Zápis této

[1] Platí to i pro singulární čísla. Pro přehlednost však budou dále uváděny jen vlastní čísla.

techniky vyjadřuje rovnice 5.

$$\lambda_k > \frac{1}{n}\sum_{k=1}^{n} \tag{5}$$

Experiment 7.1 se zabývá volbou optimálního počtu vlastních čísel a zda je lepší použít „elbow“ techiku či techniku založenou na průměru.

5.2.2 Kroky metody

1. **Výběr n-dimenzionálních dat**
   Vybereme data, na kterých lze metodu PCA spustit, např. obrázek reprezentovaný maticí.

   Data bychom si měli před spuštěním metody upravit. Můžeme ignorovat konstanty, protože mají nulový rozptyl a nepomohou k relevantnějším výsledkům. Další parametry, které můžeme vyloučit, jsou takové, které mají velmi malý rozptyl. Jako poslední lze ignorovat parametry, které jsou lineárně závislé na jiných parametrech.

   Naopak parametry vhodné pro ponechání jsou takové, které buďto nejsou závislé na jiných parametrech a nebo takové, které mají vysoký rozptyl.

2. **Nalezení průměru každé dimenze dat**
   Po předzpracování dat je nutné nalézt průměr každé dimenze ze zadaných dat. Tedy:

   $$\overline{x} = \frac{\sum_{i=1}^{10} x_i}{10} \tag{6}$$

   Po odečtení průměrů od původních dat dostaneme data, která mají nulový průměr.

   V následujícím příkladu jsou tři parametry X, Y a Z, z toho vyplývá, že počet dimenzí bude $n = 3$. Dále má každý parametr určitý počet hodnot, v tomto příkladu 10. Data lze vidět v tabulce 1 a na obrázku 2. Nejprve se získá průměr každé dimenze (parametru) a následně na základě průměru budou vytvořena data nová, pro která platí, že $x' = x - \overline{x}$ a $\mu' = 0$. Nová data jsou zobrazena v tabulce 2 a na obrázku 3.

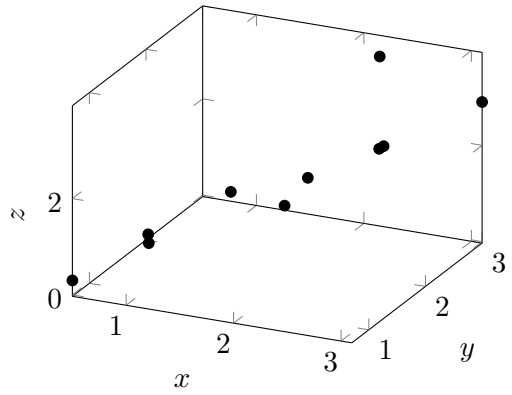
3. **Výpočet kovarianční matice**
   Dalším krokem je výpočet kovarianční matice. Kovarianční matice byla vysvětlena v kapitole 2. Na základě počtu dimenzí n bude kovarianční matice rozměrů $n \times n$.

   $$cov = \begin{pmatrix} 0.6165556 & 0.6154444 & 0.6140111 \\ 0.6154444 & 0.7165556 & 0.7086778 \\ 0.6140111 & 0.7086778 & 0.8985878 \end{pmatrix}$$

   Důležité je, že z kovarianční matice lze vyčíst další informace. Například pokud jsou prvky neležící na diagonále nezáporné, lze říci, že se budou všechny proměnné navyšovat společně.

Tabulka 1: Původní data

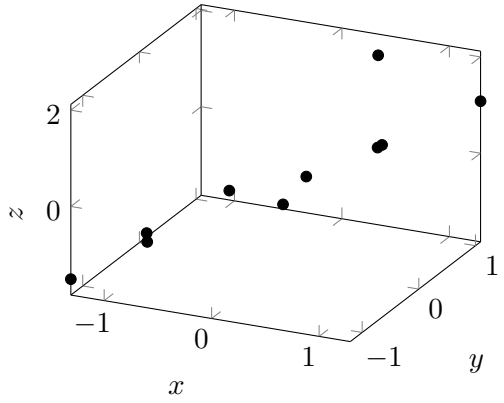
X	Y	Z
2.5	2.4	2.3
0.5	0.7	0.3
2.2	2.9	3.6
1.9	2.2	1.6
3.1	3.0	2.9
2.3	2.7	1.9
2	1.6	1.6
1	1.1	0.9
1.5	1.6	1.7
1.1	0.9	1.3



Obrázek 2: Vizualizace původních dat

Tabulka 2: Původní data s nulovým průměrem

$X - \overline{X}$	$Y - \overline{Y}$	$Z - \overline{Z}$
0.69	0.49	0.49
-1.31	-1.21	-1.51
0.39	0.99	1.79
0.09	0.29	-0.21
1.29	1.09	1.08
0.49	0.79	0.09
0.19	-0.31	-0.21
-0.81	-0.81	-0.91
-0.31	-0.31	-0.11
-0.71	-1.01	-0.51



Obrázek 3: Vizualizace původních dat s nulovým průměrem

4. **Výpočet vlastních vektorů a jejich vlastních čísel**

   Vlastní vektory a čísla jsou velmi důležitým prvkem při použití metody PCA. Stejně jako kovarianční matice, i vlastní vektory a čísla byly popsány v kapitole 2.

$$vlastniCisla = \begin{pmatrix} 2.04831131 \\ 0.14001186 \\ 0.04337571 \end{pmatrix}$$

$$vlastniVektory = \begin{pmatrix} -0.5185836 & -0.5931106 & 0.6158660 \\ -0.5759394 & -0.2900574 & -0.7643040 \\ -0.6319532 & 0.7510569 & 0.1911768 \end{pmatrix}$$

Pokud bychom proložili data vlastními vektory, zjistili bychom, že významějším vektorem

je vektor (-0.5185836; -0.5759394; -0.6319532), protože se okolo něj shromažďuje většina dat.

5. **Výběr komponent pro redukci dimenze a vytvoření matice příznakových vektorů**

   Jakmile získáme vlastní vektory z kovarianční matice, je nutné je uspořádat podle vlastních čísel od největšího po nejmenší. To znamená, že nyní máme hlavní komponenty (vlastní vektory) seřazeny podle významnosti. Nyní se můžeme rozhodnout, které hlavní komponenty použijeme a které nikoliv. Například můžeme vynechat takové hlavní komponenty, které mají nízkou významnost, tedy nulovou či velmi blízkou nule. Ano, přijdeme sice o některé informace, ale ne o tolik, aby to bylo podstatné pro analýzu těchto dat. Tím, že odebereme jednu komponentu, redukujeme data o jednu dimenzi. Obecně, pokud budeme mít n-dimenzionální data a vybereme pouze prvních k vlastních vektorů, budou mít výsledná data k dimenzí.

   Nyní zbývá vytvořit matici příznakových vektorů. Vytvoříme novou matici na základě vybraných vlastních vektorů, které chceme použít.

   Ze získaných vektorů vybereme tolik důležitých vektorů, jakou chceme mít velikost redukované dimenze. Data jsou 3-dimenzionální, a proto bude následovat ukázka s jedním i se dvěma nejvýznamnějšími vektory.

$$vlastniVektory = \begin{pmatrix} -0.5185836 & -0.5931106 & 0.6158660 \\ -0.5759394 & -0.2900574 & -0.7643040 \\ -0.6319532 & 0.7510569 & 0.1911768 \end{pmatrix}$$

   Nejvýznamnější vektor (dle vlastních čísel) pro redukci do 1D je:

$$priznakoveVektory = \begin{pmatrix} -0.5185836 \\ -0.5759394 \\ -0.6319532 \end{pmatrix}$$

   V případě redukce do 2D jsou významnější následující dva vektory:

$$priznakoveVektory = \begin{pmatrix} -0.5185836 & -0.5931106 \\ -0.5759394 & -0.2900574 \\ -0.6319532 & 0.7510569 \end{pmatrix}$$

   Data budou redukována buďto o jednu, či dvě dimenze.

6. **Získání „nových“ dat**

   Chceme získat data nová, otázka zní „Jak?“. Postačí k tomu matice příznakových vektorů a data s nulovým průměrem. Výsledná rovnice bude vypadat následovně:

$$NovaData = Data * PriznakoveVektory \tag{7}$$

   PriznakoveVektory je matice vybraných vlastních vektorů. Proměnná *Data* vyjadřuje data s nulovým průměrem. *NovaData* je matice vyjadřující výsledek projekce.

   Výsledná data při redukci za použití dvou vlastních vektorů jsou prezentována na obrázku 5.

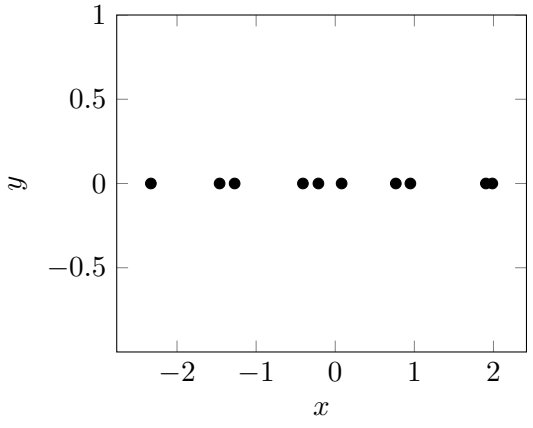
   V případě použití jednoho vektoru budou data ležet na přímce. Výsledek při použití jednoho vlastního vektoru lze vidět na obrázku 4 .

Tabulka 3: Nová data při použití jednoho vlastního vektoru (redukce 3D do 1D)

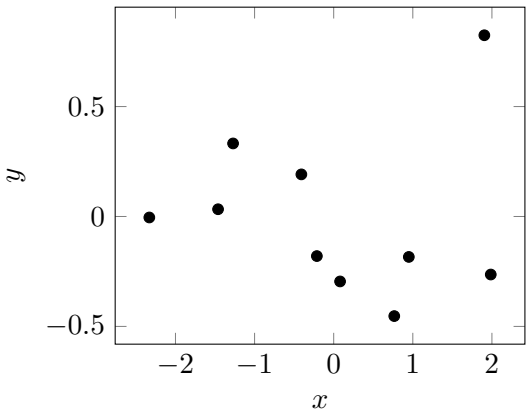
X
0.94959
-2.33053
1.90421
0.08070
1.98542
0.76555
-0.21277
-1.46163
-0.40866
-1.27187

Tabulka 4: Nová data při použití dvou vlastních vektorů (redukce 3D do 2D)

X	Y
0.94959	-0.18396
-2.33053	-0.00451
1.90421	0.82431
0.08070	-0.29546
1.98542	-0.26400
0.76555	-0.45301
-0.21277	-0.18006
-1.46163	0.032980
-0.40866	0.19150
-1.27187	0.33221



Obrázek 4: Vizualizace výsledných dat po redukci do 1D



Obrázek 5: Vizualizace výsledných dat po redukci do 2D

5.2.3 Získání aproximovaných dat

V případě obrázků chceme získat aproximovaná data. V následujících rovnici jsou použity proměnné se stejným významem jako v rovnici 7. Rovnici pro získání aproximovaných dat lze zapsat takto:

$$RedukovaneData = (NovaData \times PriznakoveVektory^T) + Prumer$$

Proměnná *RedukovaneData* zastupuje matici s transformovanými daty. *Prumer* je proměnná obsahující průměr, který byl vypočítán v bodě 2.

Tento postup PCA lze algoritmicky zapsat takto:

Algoritmus 1 Algoritmus PCA

Vstup: Reálná matice A o rozměrech $m \times n$, celé číslo k

Výstup: Matice A' o rozměrech $m \times k$

1: Spočítej průměr z každé dimenze matice A.
2: Spočítej kovarianční matici.
3: Spočítej vlastní čísla a k nim příslušné vlastní vektory.
4: Seřaď vlastní vektory dle významnosti a vyber k prvních vlastních vektorů.
5: Udělej projekci vzorků do nového podprostoru pomocí vybraných vlastních vektorů.
6: Vrať výsledek projekce.

5.3 Kernel PCA

Kernel PCA je rozšířením výše zmíněné metody PCA. Využívá tzv. kernel metod, jejichž výhodou je, že umožňují metodě PCA provést nelineární dimenzionální redukci a ne pouze lineární, pro kterou je původně určena. Metoda PCA pracuje pouze s lineárně oddělitelnými daty čili takovými, že pokud prostor $\mathbb{R}^2$ proložíme přímkou, na jedné straně se budou nacházet významější komponenty a na druhé ty méně významné. Kernel PCA umožňuje pracovat s daty, která nejsou tzv. lineárně oddělitelná. Pro představu jsou to taková data, která nelze proložit přímkou, ale například kružnicí.

Pokud tedy chceme provést nelineární dimenzionální redukci pomocí metody PCA, je vhodné ji rozšířit o kernel metodu, která promítne lineárně neoddělitelná data do prostoru, kde již mohou být lineárně oddělitelná a metoda PCA s nimi může bez obav pracovat. Máme tedy metodu pro zobrazení nelineárních prvků na lineární, nazvěme ji např. ϕ. Kernel metoda vypočítá skalární součin obrazů vzorků x pro zobrazení ϕ.

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

Ve zkratce kernel metoda ϕ mapuje data z původního n-dimenzionálního prostoru do obvykle většího k-dimenzionálního prostoru tím, že vytvoří nelineární kombinace původních vlastností.

Nevýhoda výpočtu skalárního součinu je, že čím více bude prvků ve vzorku a čím větší bude počet dimenzí, tím nákladnější tento výpočet bude. Proto je vhodné použít tzv. kernel trik,

který snižuje náročnost výpočtu, že místo toho, aby se počítaly souřadnice jednotlivých prvků z dat, počítá se pouze vzdálenost mezi obrazy každé dvojice dat.

Kernelů je velké množství [39], popsány budou alespoň ty nejvíce populární - lineární, polynomiální a gaussovský.

5.3.1 Lineární kernel

Lineární kernel je typ kernelu, který bude s použitím metody PCA vydávat stejné výsledky jako metoda PCA bez tohoto kernelu.

5.3.2 Polynomiální kernel

Polynomiální kernel je vhodný pro normalizované soubory dat. Je ho možné zapsat takto:

$$k(x, y) = (x^T y + c)^d$$

Parametr d je stupeň polynomiálního kernelu. Pokud bude např. $d = 2$, bude kernel kvadratický apod.

Nevýhodou oproti dvěma zmíněným kernelům je, že nemusí poskytovat tak vysokou přesnost. Převážně však až ve vyšších dimenzích.

Polynomiální kernel se stal účinným nástrojem při zpracování přirozeného jazyka [40].

5.3.3 Gaussovský kernel

Posledním kernelem, který je třeba zmínit, je gaussovský kernel. Tento kernel je velice populární pokud jde o použití s PCA. Jeho hlavní předností je, že dokáže mapovat do nekonečně dimenzionálního prostoru [41]. Tohle zjištění je však třeba brát pouze jako skutečnost a nesnažit se jej používat v praxi, počítače přece jen mohou pracovat pouze s konečnými věcmi a nemohou počítat donekonečna. Avšak nekonečně dimenzionální prostor je, i v teoretické rovině, někdy zase nutné převést zpět do konečně dimenzionálního prostoru a to tak, že se zvolí libovolné konečné číslo, které bude vyjadřovat velikost dimenze. Gaussovský kernel se zapisuje takto:

$$k(x, y) = exp(-\sigma||x - y||^2)$$

Velkou roli zde hraje parametr σ a je třeba jej pečlivě volit. Parametr σ značí šířku kernelu nebo rozsah, který pokrývá. Pokud bude parametr přeceněn, bude exponenciála téměř lineární a tím klesne i síla gaussovského kernelu ve vyšších dimenzích. Pokud bude naopak podceněn, může být rozhodování ohledně hranic vysoce citlivé.

Oproti lineárnímu kernelu je vhodné použít gaussovský kernel v případě, že máme menší počet vlastností, například několik tisíc a velikost vzorku středně velkou, například desítky tisíc prvků.

Obecný algoritmus pro kernel PCA vypadá následovně:

Algoritmus 2 Algoritmus KPCA

Vstup: Matice $A \in \mathbb{R}$ o rozměrech $m \times n$, celé číslo k

Výstup: Matice A' o rozměrech $m \times k$

1: Spočítej kernel matici pomocí libovolné kernel metody.
2: Spočítej z kernel matice vlastní čísla a k nim příslušné vlastní vektory.
3: Seřaď vlastní vektory dle významnosti a vyber k prvních vlastních vektorů.
4: Udělej projekci vzorků do nového podprostoru pomocí vybraných vlastních vektorů.
5: Vrať výsledek projekce.

5.4 CUR

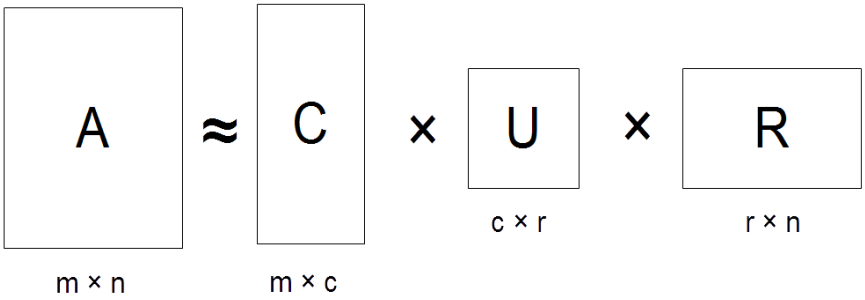
Dříve než bude zmíněna metoda CUR samotná, je potřeba popsat důvody jejího vzniku. Nejprve je nutné zmínit, že CUR není, jako ostatní vybrané metody, metoda pro redukci dimenzionality. CUR je aproximace, kterou lze použít namísto metody SVD s nízko-úrovňovou aproximací. Proč je CUR lepší než metoda SVD je zdůvodněno v následujících řádcích.

SVD má jako každá metoda své problémy, ty je však možné řešit pomocí CUR. Prvním problémem je, že SVD narušuje strukturu dat. Pokud je vstupní matice řídká, SVD vytvoří matici hustou a díky ortogonalizaci tím snižuje kvalitu reprezentace výsledných dat. Podobný problém, který snižuje kvalitu reprezentace výsledých dat, může být to, že pokud máme data obsahující pouze nezáporné vektory, nelze zaručit, že během aproximace budou tyto nezáporné hodnoty zachovány. Mezi problémy patří také špatná reprezentace toho, co nově vzniklé souřadnice znamenají.

CUR se snaží nenarušovat strukturu dat a vytvářet takové výsledky, které budou pro člověka lépe uchopitelné oproti těm, které vydává SVD. CUR je nízkoúrovňová aproximace vyjádřená v malém počtu řádků a sloupců původní matice A. Tím, že vybereme pouze určité řádky a sloupce, můžeme usnadnit jejich reprezentaci.

CUR vybírá určitý počet řádků r a určitý počet sloupců c z matice A, která je o rozměrech $m \times n$. Vybrané řádky a sloupce mohou být pak přesně aproximovány jako vážený produkt sloupců a řádků původní matice. Z této matice CUR vytvoří tři matice - C, U a R. Matice C o rozměrech $m \times c$ je matice c sloupců z matice A, podobně R je matice o rozměrech $r \times n$, která obsahuje r řádků z matice A. Matice U se nazývá maticí vah a její rozměry jsou $c \times r$. Matice U může být spočítána jako C^+AR^+, kde C^+ a R^+ jsou pseudoinverzními maticemi k C, resp. R. Výsledná matice $A' = CUR$ se pak nazývá CUR aproximací matice A. Aproximaci matice A lze vidět na obrázku 6

Výše uvedené lze přepsat do následujícího algoritmu:



Obrázek 6: Aproximace matice A pomocí metody CUR

Algoritmus 3 Základní algoritmus CUR

Vstup: Matice $A \in \mathbb{R}$ o rozměrech $m \times n$, celé číslo c a celé číslo r

Výstup: Matice C o rozměrech $m \times c$, matice U o rozměrech $c \times r$ a matice R o rozměrech $r \times n$

1: Vytvoř matici C tak, že vybereš c sloupců z matice A.
2: Vytvoř matici R tak, že vybereš r řádků z matice A.
3: Vytvoř matici U, která je rovna C^+AR^+, kde C^+ a R^+ jsou pseudoinvezními maticemi matic C a R.

Stejně jako u ostatních metod i zde je problém s určením správného počtu řádku a sloupců. Mohli bychom samozřejmě vybrat všechny kombinace řádků a sloupců, ale to by bylo výpočetně náročné a navíc zbytečné. Nabízí se vícero možností, například použití tzv. deterministického CUR nebo použít algoritmus náhodného výběru.

5.4.1 Algoritmy náhodného výběru sloupců a řádků

Jednotlivé řádky (či sloupce) obodujeme pomocí určitého skóre a k nim přidáme sloupce, které odpovídají tomuto obodování. K výpočtu skóre se většinou používají dvě techniky - euklidovská norma (CUR-L2) a statistické vyhledávání vlivných bodů (CUR-SL).

Algoritmus CUR-L2 nelze použít na normalizované sloupce a řádky. Z normálního rozdělení by vzniklo rozdělení rovnoměrné. Algoritmus CUR-SL není vhodný pro velmi velké matice, neboť výpočet nejlepších k singulárních vektorů roste kvadraticky s k a lineárně s velikostí sloupců a řádků. Nejdůležitějším nedostatkem je však to, že výsledná přesnost aproximace je často mnohem nižší než u metody deterministického CUR.

Algoritmus pomocí euklidovské normy lze vyjádřit takto:

Algoritmus 4 Algoritmus CUR-L2:

Vstup: Reálná matice A o rozměrech $m \times n$, celé číslo c/r

Výstup: Matice $W \in \mathbb{R}$ o rozměrech $m \times c/r \times n$

1: Pro $l = 1...n$ dělej: $P(l) = \sum_i (X_{i,l}^2 / \sum_{i,j} X_{i,j}^2)$

2: Pro $i = 1...k$ dělej: Náhodně vyber $W_{*,i} \in X$ podle $P(l)$

5.4.2 Deterministický CUR

Výběr sloupců pro aproximaci je spojen s problémem CSSP [2]. Pro metodu CUR bylo vymyšleno několik algoritmů, které jsou založeny na deterministickém přístupu pro výběr sloupců/řádků [26]. Většina z těchto algoritmů funguje velice dobře na malých datových sadách. Vzhledem k tomu, že jejich výpočet je velice komplexní, nepoužívají se tyto algoritmy na velké matice.

Výběr vhodné podmnožiny sloupců pro problém CSSP spočívá v tom, že se hledá největší determinant matice C, která obsahuje k vybraných sloupců. To samé platí i pro výběr řádků. Důvodem proč volit hladový či deterministický algoritmus je, že problém výběru maximálního objemu je NP-těžký a proto ho běžnými algoritmy nelze řešit.

Tato podkapitola měla pouze uvést v povědomí to, že existují algoritmy, které se snaží vybírat počet sloupců a řádků deterministickým či hladovým přístupem. O jednom z nich, tzv. LS-DCUR (Large scale - Deterministic CUR) se lze více dozvědět v této vědecké práci [26]. Algoritmus zvládá i poměrně velké matice a je tedy nejspíše nejvhodnějším deterministickým algoritmem pro CUR. Konkrétní příklady použití však budou používat pouze klasický algoritmus CUR, případně pro optimálnější výběr bude použit algoritmus výběru sloupců na základě skóre.

Mimo výše dva zmíněné algoritmy, které zlepšují výpočet CUR, existuje ještě několik algoritmů. Tyto algoritmy vylepšují relativní chybovost algoritmu. Mezi takové algoritmy se řadí například CUR pomocí adaptivního vzorkování [42] či tzv. „Rychlý algoritmus CUR“ [43].

5.5 NMF

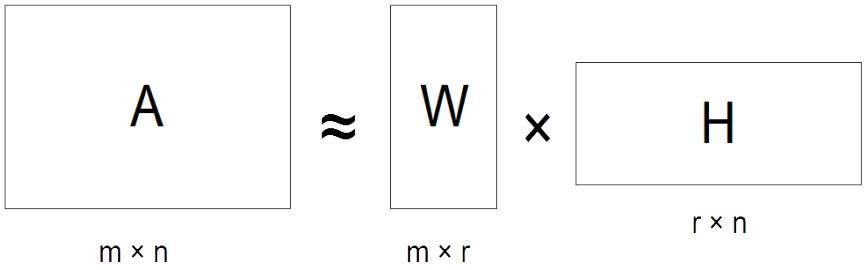
Slovo „Non-Negative“ naznačuje, že metoda NMF pracuje s maticemi, které neobsahují záporné prvky. Vhodnou maticí pro použití NMF je například matice termů-dokumentů, která v řádcích reprezentuje jednotlivé dokumenty a ve sloupcích počet výskytů jednotlivých slov v daném dokumentu, kde počet rozhodně nemůže být záporné číslo. Díky nezáporosti výsledků se i získaná data lépe analyzují.

Metoda NMF rozkládá původní matici A na dvě matice - matici faktorů W a směšovací matici H. Rovnice pro NMF vypadá následovně:

$$A = WH \text{ a platí } ||A - WH|| \leq \epsilon \text{ pro dané } r$$

Matice A má rozměry $m \times n$, matice faktorů W $m \times r$ a matice H má rozměry $r \times n$. Rozklad matice A pomocí metody NMF znázorňuje obrázek 7.

[2] Z anglického Column subset selection problem.



Obrázek 7: Rozklad matice A pomocí metody NMF

Parametr r je právě použit pro aproximaci, jak lze vidět na rovnici 8. Většinou je volen tak, že $(n+m)r \leq nm$. Součin matic W a H není přímo roven původní matici A, tento součin nazýváme přibližnou aproximací dle určité hodnoty, často označován jako r. Tím, že WH je určitou aproximací matice A, lze tvrdit, že WH v sobě nese komprimovaná data matice A. NMF využívá minimalizujících numerických metod a díky nim dokáže získat určité vlastnosti, které jsou uloženy v matici W. Tyto vlastnosti posléze usnadňují identifikaci a klasifikaci. Pro lepší představu lze vlastnostmi nazvat například části textových dokumentů či obrázků.

$$A_{m,n} = \sum_{i=1}^{r} W_{m,i} H_{i,n} \tag{8}$$

K nalezení vhodné aproximace metody NMF existuje několik řešení. Je třeba najít funkci, která vyjádří kvalitu zvolené aproximace. Může jí být frobeniova norma rozdílu matic, která je vyjádřena rovnicí 9.

$$\|A - B\|^2 = \sum_{ij} (A_{ij} - B_{ij})^2 \tag{9}$$

Další metodou pro měření kvality může být divergence z A do B 10.

$$D(A\|B) = \sum_{ij} (A_{ij} \log \frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij})^2 \tag{10}$$

K algoritmu pomocí násobícího pravidla je zapotřebí mít následující tři rovnice:

$$W_{ij} = \sum_{k=1}^{m} \frac{A_{ik}}{(WH)_{ik}} H_{jk} \tag{11}$$

$$W_{ij} = \frac{W_{ij}}{\sum_{k=1}^{n} W_{kj}} \tag{12}$$

$$H_{ij} = \sum_{k=1}^{n} W_{ki} \frac{A_{kj}}{(WH)_{kj}} \tag{13}$$

Za využití těchto rovnic je možné sestavit algoritmus pro metodu NMF.

Algoritmus 5 Algoritmus NMF

Vstup: Matice A o rozměrech $m \times n$ a $r \in \mathbb{N}$

Výstup: Matice W o rozměrech $m \times r$ a matice H o rozměrech $r \times n$

1: Náhodně vygeneruj počáteční hodnoty pro W a H a nastav $k = 1$
2: **while** WH není konvergentní **do**
3: Aktualizuj báze matice W pomocí rovnic 11 a 12
4: Aktualizuj koeficienty matice H pomocí rovnice 13
5: $k = k\ + 1$
6: **end while**

Algoritmus pomocí násobícího pravidla je pomalejší než další dva algoritmy, zmíněné později, z toho důvodu, že vyžaduje více náročnějších iterací. Dalším problémem je, že tento algoritmus konverguje velice pomalu. Existují modifikace algoritmu, které zrychlují výpočet, avšak neřeší problém pomalé konvergence a na druhou stranu existují i takové modifikace, které řeší problém konvergence na úkor rychlosti. Při použití algoritmu pomocí násobícího pravidla je třeba zvážit rychlost algoritmu a rychlost konvergence.

Mezi další algoritmy pro NMF patří algoritmus pomocí gradietního sestupu nebo algoritmus pomocí metody nejmenších čtverců. Další zmiňované algoritmy jsou rychlejší, avšak algoritmus pomocí gradientního sestupu neřeší problém pomalé konvergence, tento problém řeší algoritmus pomocí metody nejmenších čtverců. Pro experimenty v této práci bude použit algoritmus pomocí násobicího pravidla, kde bude konvergence měřena frobeniovou normou rozdílu matic.

6 Implementace

Tato kapitola popisuje implementaci vybraných metod. Byly vytvořeny dva typy aplikací - pro obrázky a dokumenty. Dále zde bude uveden popis architektury aplikací, popis jednotlivých komponent, jejich tříd a metod. Před popisem samotné implementace bude vysvětlena práce s datovými kolekcemi a použitými knihovnami.

6.1 Desktopové aplikace

Pro snadnější obsluhu naimplementovaných aplikací pro redukci byly vytvořeny dvě desktopové aplikace s grafickým uživatelským rozhraním (GUI). GUI bylo zvoleno z toho důvodu, aby uživatel nemusel obrázky dohledávat ve svém souborovém systému nebo nemusel číst výsledky volaných metod z konzole, která je graficky hodně omezena. Pro implementaci byl vybrán .NET framework, konkrétně jazyk C#. GUI bylo tvořeno pomocí technologie WinForms. Detailněji jsou tyto aplikace popsány v přílohách B.1 a B.2.

6.2 Použité datové kolekce pro experimenty

Pro experimenty byly vybrány dvě datové kolekce, a to obrázky a dokumenty. Pro použití těchto kolekcí bylo třeba, aby data byly reprezentována jako matice, s kterou by mohly metody redukcí dimenzionality pracovat.

6.2.1 Obrázky

Sestavení matice pro obrázek se liší v tom, jestli požadujeme obrázek ve stupních šedi či barevný. U barevného obrázku záleží, které vlastnosti kromě RGB uchovávat. Mezi těmito vlastnostmi může být alfa kanál (průhlednost), jas, saturace nebo zabarvení.

Mějme obrázek o rozlišení 180 pixelů $\times$ 180 pixelů a chceme získat obrázek ve stupních šedi reprezentovaný maticí. Jediné co je potřeba udělat, je vytvořit matici o stejných rozměrech jako má obrázek. Postup při tvorbě matice pro obrázek ve stupních šedi, kdy na vstupu předpokládáme obrázek barevný, je následující:

1. Převeď obrázek na stupně šedi
2. Pro každý pixel získej hodnotu červené, zelené a modré barvy na i-té a j-té pozici obrázku.
3. Sečti získané barevné hodnoty a vyděl je třemi.
4. Tuto hodnotu ulož do matice na i-tou a j-tou pozici a nastav ji všem barevným složkám konkrétního pixelu v obrázku.

Zpětná konverze matice na obrázek je obdobná. Získá se hodnota šedi z matice a v novém obrázku na stejné pozici jako v matici se vytvoří pixel, jehož jednotlivé RGB složky budou odpovídat této hodnotě.

U barevných obrázků je maticová reprezentace trochu složitější. Matice pro barevný obrázek stejných rozměrů jako v předchozím příkladu nebude 180×180, ale bude se lišit dle zvolených vlastností, které je třeba zachovat. Minimálně však bude mít rozměry $180 \times 3 * 180$ z toho důvodu, že se zachovávají v tom nejzákladnějším případě všechny tři barevné složky RGB. Obecně tedy bude mít matice rozměry $180 \times n * 180$, kde n je počet zachovaných vlastností obrázku.

Konverze barevného obrázku na matici je realizována tak, že program přečte n požadovaných vlastností pixelu na i-té a j-té pozici v obrázku s tím, že $n \geq 3$. Požadované vlastnosti následně ukládá do matice tak, že první vlastnost uloží na i-tou a j-tou pozici matice, další na i-tou a j+1 pozici matice, atd. Poslední vlastnost uloží na i-tou a (j+n-1)-tou pozici matice. Další pixel v řádku bude ukládán na i-tou a j+n-tou pozici matice. Poslední vlastnost posledního pixelu zapíše na i-tou pozici a j+m*n-1 pozici v matici, kde m je počet pixelů v jednom řádku obrázku. Při skoku na další řádek pixelů v obrázku opět opakuje tento postup, s výjimkou, že $i = i + 1$.

Zpětná konverze matice na barevný obrázek je obdobná. Z matice vyčte n požadovaných vlastností a nastaví je pixelu na konkrétní pozici.

Obrázky pro experimenty jsou rozděleny do tří kategorií - malé, střední a velké obrázky. Do kategorie pro malé obrázky spadají obrázky s počtem pixelů v rozsahu deset až sto tisíc. Pro střední je tento rozsah sto tisíc až jeden milión. Pro velké jeden milión a více pixelů.

Barevné obrázky jsou pouze alternativou k obrázkům ve stupních šedi. Pro experimenty byly z důvodu rychlejšího výpočtu použity obrázky ve stupních šedi. V případě barevných obrázků je komprese stejná, ale navíc je výsledný obrázek barevný.

6.2.2 Dokumenty

Reprezentace dokumentů bývá obvykle realizována pomocí matice termů-dokumentů nebo matice TF-IDF. V řádcích matice termů-dokumentů jsou jednotlivé dokumenty a ve sloupcích slova, která se v těchto dokumentech vyskytují. Pokud je ve sloupci například slovo „diplomová“ a řádek je libovolný dokument, tak průsečík tohoto řádku a sloupce značí počet výskytů slova v tomto dokumentu.

Postup vytvoření matice termů-dokumentů je následující:

1. Získej všechny slova z dokumentů, která nejsou stopslova. Stopslova jsou slova, která nemají pro analýzu velkou informativní hodnotu i přesto, že se mohou vyskytovat v dokumentech velmi často.

2. Sestav matici $m \times n$, kde m je počet slov (termů) a n počet dokumentů.

3. Jednotlivým buňkám matice nastav číslo, které bude odpovídat počtu výskytu slova i-tého slova v j-tém dokumentu, kde i je aktuální řádek a j je aktuální sloupec matice.

Další možnou reprezentací dokumentů je TF-IDF (term frequency - inverse document frequency) matice. K vytvoření této matice postačí znalost dvou rovnic, pro část TF a pro část IDF.

TF vyjadřuje počet výskytů slova v jednom dokumentu dělený počtem výskytů slova ve všech dokumentech. Vzhledem k tomu, že se však TF-IDF matice může vytvářet pro všechna slova, tak TF může přiřazovat velkou váhu slovům, které se v sadě dokumentů vyskytují dost často. Proto je třeba tuto váhu usměrnit a k tomu právě slouží IDF.

$$TF = \frac{počet\ výskytů\ slova\ v\ dokumentu}{počet\ výskytů\ slova\ ve\ všech\ dokumentech} \tag{14}$$

IDF nahlíží na prvky s vysokou váhou jako na méně důležité než ty, které se vyskytují ojediněle. IDF vyjadřuje důležitost konkrétního slova.

$$IDF = \log(\frac{celkový\ počet\ dokumentů}{počet\ dokumentů\ obsahujících\ dané\ slovo}) \tag{15}$$

Výslednou hodnotu TF-IDF pro dané slovo získáme vynásobením TF hodnoty s IDF hodnotou [44].

$$TFIDF = TF \times IDF \tag{16}$$

Tato práce se zabývá i tím, zda je pro konkrétní aplikace vhodnější matice termů-dokumentů nebo matice TF-IDF.

Pro práci byla vybrána kolekce BBC [45], poskytující přes dva tisíce dokumentů v pěti různých tématech (ekonomika, zábava, politika, sport, technika).

6.3 Použité knihovny

Tato práce využívá jedné knihovny a jednoho open source projektu. Knihovna Accord.NET poskytuje velké množství matematických operací a blíže je vysvětlena v kapitole 6.3.1. Pro vytvoření TF-IDF matice ze sady dokumentů byl použit a mírně upraven projekt Koryho Beckera. Krátké seznámení s tímto projektem poskytuje kapitola 6.3.2.

6.3.1 Accord.NET

Accord.NET [46] je framework pro strojové učení a slogan tohoto frameworku zní *„Machine learning made in a minute“*. Poskytuje řešení pro velké množství problémů a díky velmi vhodně strukturované architektuře a intuitivnímu pojmenování funkcí a knihoven je mnoho těchto problémů vyřešeno velmi rychle.

Knihovna se skládá ze čtyř modulů:

- Accord.Math - modul pro maticovou algebru či rozklad matic.
- Accord.Statistics - modul, který umožňuje provádět statistické výpočty, ale například i výpočet metod PCA nebo KPCA, vizualizaci do histogramu či grafu a mnoho dalšího.

- Accord.MachineLearning - modul poskytující například SVM (Support Vector Machines), rozhodovací stromy nebo k-means.
- Accord.Neuro - modul zaměřený na neuronové sítě a s nimi spojené techniky.

Knihovna Accord.NET byla zvolena hlavně z toho důvodu, že poskytuje komplexní řešení pro problematiku této práce. Využity byly první dva moduly - Math a Statistics. Modul Math byl využit pro základní operace nad maticemi, singulární rozklad a pro získání vlastní čísel a vektorů. Mimo tyto metody byla využita u zpracování dokumentů třída *NonnegativeMatrixFactorization* z důvodu, že vlastní implementace NMF byla ve srovnání s implementací NMF z Accord.NET pomalejší. Na kompresi obrázků však tato třída využita není.

U modulu *Statistics* bylo využito především jeho statické třídy *Tools*, která poskytuje nástroje pro výpočet mnoha prvků souvisejících se statistikou. Mezi ty, které byly využity v této práci, patří výpočet průměru, kovarianční matice a korelační matice. Dále byla využita pro experimenty s dokumenty i třída *PrincipalComponentAnalysis* z jmenného prostoru *Analysis*. Důvodem jejího využití byla větší rychlost nad velkými daty.

Jedinou metodou, která ve sbírce knihovně Accord chybí, je metoda pro rozklad matic CUR.

Rychlost knihovny je poměrně dobrá i při větších datech. Velkým plusem je však již zmiňovaná výborná ovladatelnost a komplexnost knihovny.

6.3.2 TF-IDF

Tento projekt [47] pomohl sestavit na základě sady dokumentů TF-IDF matici normalizovanou pomocí euklidovské normy. Výsledná matice byla následně použita v experimentech s dokumenty. Struktura projektu je velmi jednoduchá a jeho začlenění do existujícího systému taktéž. Projekt je možné nalézt v adresáři *TFIDFExample*.

Autor projektu nadefinoval nemalou sadu stopslov, kterou lze naléz ve třídě *StopWords.cs*. Tuto sadu je však možné změnit, jako kteroukoliv část projektu. Předdefinovaná stopslova zahrnují běžně používané spojky, zájmena, příslovce, ale taky několik přídavných jmen či sloves.

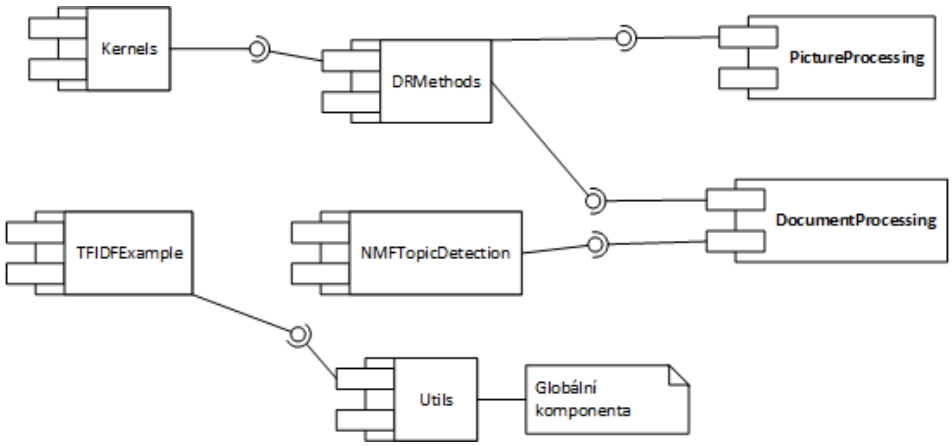
Hlavní třídou je však *TFIDF.cs*, která kromě privátních pomocných metod poskytuje navenek následující metody:

- Transform - hlavní metoda pro transformaci sady dokumentů na matici TF-IDF. Na vstupu požaduje sadu dokumentů a nadefinovat tzv. *treshold*, který je ve výchozím stavu nastaven na 3. Na základě tresholdu vybírá slova, jejichž počet výskytů je větší nebo roven určenému tresholdu.
- Normalize - normalizuje výslednou matici pomocí euklidovské normy.
- Save - metoda pro uložení slovníku do datového souboru. Slovníkem je myšleno slovo a jeho IDF hodnota.
- Load - metoda pro načtení slovníku z datového souboru.

Výhodou použití tohoto open source projektu byla plná kontrola nad zdrojovým kódem projektu a z toho plynoucí jednoduchá modifikovatelnost zdrojového kódu. Modifikace byla však nutná pouze jednou, při potřebě získání množiny slov (termů) použitých v matici TF-IDF.

6.4 Implementace vlastních řešení a jednotlivých aplikací

Praktická část diplomové práce je rozdělena pro přehlednost do několika komponent. Rozvržení systému a závislosti komponent na sobě zobrazuje diagram komponent na obrázku 8. Na komponentě *Utils* je závislá většina ostatních komponent, kromě komponenty *TFIDFExample*. Z toho důvodu byla tedy pro přehlednost diagramu označena jako „globální komponenta“.



Obrázek 8: Diagram komponent

6.4.1 Komponenta DRMethods

Nejdůležitější komponentou pro celou diplomovou práci je komponenta DRMethods, která poskytuje implementaci jednotlivých metod redukcí dimenzionality.

SingularValueDecomposition

Třída obsluhující volání metod třídy *SingularValueDecomposition* z knihovny Accord.NET, zajišťující provádění nízkoúrovňové aproximace nebo výpočet LSI.

Nezávisle na použití je třeba nejprve pomocí konstruktoru *SingularValueDecomposition(double[,] A, bool useTechnique, double elbowRate, TechniqueType techniqueType, bool computeDecomposition)* předat některé parametry získáné z desktopové aplikace. Mezi tyto parametry patří matice, nad kterou se provádí výpočet a paramter pro určení toho, zda se bude provádět výpočet pomocí techniky pro určení vhodného podprostoru. Pokud se bude provádět výpočet pomocí techniky pro určení vhodného podprostoru, předáme parametr, o jaký typ se jedná a v případě elbow techniky i to, kolik procent vlastností je potřeba ponechat. Pokud chceme zachovat poslední výpočet rozkladu matice A, bude *computeDecomposition* nastaven na *true*. Po předání všech

potřebných parametrů pro výpočet SVD je nutné zavolat metodu *Compute()*. Tato metoda vytvoří instanci třídy *SingularValueDecomposition* z knihovny Accord a uloží tři výsledné matice do instančních proměnných pro další výpočty. Navíc uloží do kolekce typu *List<double>* všechny singulární čísla pro případný výpočet vhodného parametru r pomocí techniky pro určení vhodného podprostoru.

Nyní záleží na konkrétní aplikaci. První možností je zavolání metody *ApproximateByLowRank(int r)* pro vypočtení nízkoúrovňové aproximace nad maticemi získanými pomocí metody *Compute()*, kde parametr r značí počet zachovaných singulárních čísel. Pokud je však povolena technika pro určení vhodného podprostoru, bude tento parametr pro výpočet nepodstatný.

Dalším možným postupem je zavolání metody *LSI(int r, double[] query)*, která slouží pro výpočet LSI nad konkrétní maticí reprezentující dokument. Parametr r má stejný význam jako u metody *ApproximateByLowRank(int r)*. Druhý parametr *query* značí dotaz, který je pro vypočtení LSI nezbytný. Konkrétně se LSI metodou zabývá kapitola 8.1.

Obě výše uvedené metody po dokončení výpočtu vracejí výsledek, který je předán desktopové aplikaci a ta jej vhodně prezentuje.

PrincipalComponentAnalysis

PrincipalComponentAnalysis je třída, která implementuje metodu PCA. Implementace odpovídá algoritmu 1. Nejprve se předají potřebné parametry pomocí konstruktoru *PrincipalComponentAnalysis(double[,] matrixA, int k, bool useTechnique, double elbowRate, MatrixType matrixType, TechniqueType techniqueType, bool runAgain)*. Mezi parametry patří matice pro výpočet, parametr k stanovující počet hlavních komponent, které mají být vybrány pro projekci, typ použité matice pro výpočet vlastních čísel a vektorů a nastavení, zda se bude používat technika pro určení vhodného podprostoru. Pokud bude technika pro určení vhodného podprostoru použita, je třeba zvolit typ této techniky a v případě elbow techniky zadat, kolik procent variability by mělo být zachováno. Pokud je potřeba počítat vlastní čísla vektory znovu, bude *runAgain* nastaven na *true*.

Po zadání všech parametrů se zavolá metoda *Compute(bool transformation)*. Metoda *Compute* spočítá PCA podle algoritmu 1. Tato metoda využívá privátní metody *GetReducetEigenVectors(double[,] matrix, int k)*. Metoda vrátí buďto stanovený počet vlastních vektorů podle parametru k, nebo pokud je povolena technika pro určení vhodného podprostoru, vrátí takové množství vlastních vektorů, které bylo touto technikou určeno jako optimální. Pokud je parametr *transformation* metody *Compute* nastaven na true, provádí se i zpětná transformace, která byla popsána v kapitole 5.2.3.

Po skončení metody *Compute* je předán desktopové aplikaci výsledek redukce dimenze se zpětnou transformací.

KernelPCA

Vzhledem k tomu, že kernel PCA je velmi podobné metodě PCA, nebude se implementace těchto

dvou metod až na základní počáteční kroky algoritmu příliš lišit. Při nastavování parametrů se pouze mění typ matice na typ použitého kernelu, který lze vybrat pomocí výčtového typu *KernelType*. Navíc se zde nastavují potřebné parametry pro vybraný kernel. Algoritmus se liší pouze v prvních dvou krocích, kdy se u KPCA nepočítá průměr a nepoužívá se korelační či kovarianční matice, ale rovnou se spočítá kernel matice na základě vybraného kernelu.

Dále je možné kernel matici centrovat pomocí metody *centerKernel(double[,] kernelMatrix)*.

NonnegativeMatrixFactorization

Třída *NonnegativeMatrixFactorization* implementuje vlastní řešení metody NMF, která odráží tři rovnice zmíněné v kapitole 5.5 a kde je konvergence vypočtena pomocí frobeniovy normy rozdílu matic.

Prvním krokem je, jako v předchozích metodách, nastavení parametrů pomocí konstruktoru *NonnegativeMatrixFactorization(double[,] A, int r, double epsilon, int maxIter, bool firstRun)*. Prvním parametrem je matice, nad kterou se bude provádět výpočet, následuje aproximační parametr r. Pro experimenty jsou zde další tři parametry. Prvním z nich je ϵ, míra tolerance toho, jak moc se mohou vzdálenosti matic porovnávaných pomocí frobeniovy normy lišit. Následuje parametr *maxIter*, který umožňuje nastavit maximalní počet iterací. Poslední parametr je *firstRun*, který je pro experimenty velmi užitečný. Tento parametr je vždy *true*, pokud se výpočet provádí nad konkrétní maticí (obrázkem či dokumentem) poprvé, případně je *true* i tehdy, pokud se změní hodnota parametru r. V ostatních případech je *false*. Toto opatření je důležité, neboť NMF vždy naplňuje matice W a H náhodnými čísly. Parametr *firstRun* je zde tedy proto, aby zachoval vždy stejnou matici, dokud nedostane jako parametr A matici jinou nebo dokud se nezmění parametr r. Těžko by se totiž zkoumal vliv parametr ϵ na počet iterací, pokud by metoda NMF vždy na počátku nastavila pro výpočet matice s náhodnými čísly.

Po nastavení parametrů se zavolá na instanci metoda *Compute()*, která po dokončení výpočtu vrátí aproximovanou matici podle parametru r. Výpočet spočívá v provádění tří privátních metod, dokud nekonverguje. Tyto metody jsou *FirstEquation(double[,] matrix)* 11, *SecondEquation()* 12 a *ThirdEquation((double[,] matrix))* 13. Metody odpovídají rovnicím zmíněným v kapitole 5.5.

CURMatrixApproximation

Nejprve se přes konstruktor *CURMatrixApproximation(double[,] A, double percentageOfRows, double percentageOfColumns)* nastaví potřebné parametry, kde A je vstupní matice a *percentageOfRows* a *percentageOfColumns* značí procento zachovaných řádků a sloupců.

Po tomto nastavení postačí zavolat metodu *ComputeCUR(bool random)* nebo *ComputeCURL2()*. První metoda počítá CUR tzv. hrubou sílou, jak popisuje algoritmus 3. To nastává v případě, pokud je parametr *random* roven *false*. Pokud je naopak *true*, vybírá sloupce a řádky na základě náhodných čísel získaných pomocí třídy *Random* jazyka C#. Druhá metoda *ComputeCURL2()* počítá CUR pomocí euklidovské normy. Tento postup je popsán v algoritmu 4.

Po výpočtu stačí na vytvořenou instanci zavolat metodu *GetResult()*, která vrátí výsledek násobení matic C, U a R.

MatrixType
Tento výčtový typ je používán u metody PCA pro omezení možností použitých matic pro výpočet na korelační nebo kovarianční.

KernelType
Výčtový typ, který umožňuje nastavit typ kernelu pouze na ten, který je aplikací podporován.

6.4.2 Komponenta Kernels

Komponenta Kernels obsahuje dvanáct tříd, kde každá třída odpovídá jedné kernel metodě. Každá z těchto tříd obsahuje metodu *Compute()*, kde se jako parametr vždy vyskytuje matice, nad kterou se má kernel metoda počítat. Dále se jako parametry metody *Compute* uvádí parametry konkrétní kernel metody. Například pro gaussovský kernel bude parametr jeden - σ. V případě polynomiálního kernelu budou tyto parametry dva - c a d. Jako výsledek vrací metoda *Compute* kernel matici, která slouží k výpočtu vlastních čísel a vektorů u metody KPCA.

6.4.3 Komponenta Utils

Tato komponenta poskytuje několik tříd, které obsahují podpůrné metody, například pro práci s obrázky či maticemi.

PictureUtils
PictureUtils je knihovní třída pro práci s obrázky. Mezi nabízené operace patří převedení obrázků na matici a zpět.

MatrixUtils
Nejobsáhlejší knihovní třídou v komponentě *Utils* je *MatrixUtils*. Poskytuje podpůrné metody pro práci s maticemi jako jsou například:

1. Vygenerování náhodné matice.
2. Výpočet frobeniovy normy.
3. Výpočet euklidovské vzdálenosti mezi vektory.
4. Výpočet cosinovy míry podobnosti.
5. Oříznutí matice na určitý počet řádků či sloupců
6. Určení vhodného podprostoru pomocí jedné z technik popsaných v kapitole 5.2.1.

DocumentMatrixType
Výčtový typ pro omezení typu matice reprezentující dokument na matici TF-IDF nebo matici termů-dokumentů.

CollectionUtils
Tato třída poskytuje jedinou podpůrnou metodu *Sort* na seřazení kolekce typu *Dictionary*.

DocumentProcessingUtils
Třída poskytující dvě stěžejní metody pro práci s dokumenty, vytvoření matice termů-dokumentů a obsluhu komponenty TFIDFExample pro vytvoření TF-IDF matice.

Ostatní privátní metody jsou pouze podpůrné metody pro vytvoření matice termů-dokumentů.

TechniqueType
Výčtový typ pro omezení typu techniky určující vhodný podprostor na dva možné.

6.4.4 Komponenty DocumentProcessing a PictureProcessing

Komponenty *DocumentProcessing* a *PictureProcessing* si jsou velmi podobné. Jde o aplikace s grafickým uživatelským rozhraním popsané v kapitole 6.1. Umožňují uživateli pohodlnější obsluhu aplikací vytvořených v rámci této diplomové práce. Obě komponenty obsahují vždy jednu třídu, která obsluhuje formulář. Taková třída v sobě obsahuje převážně zpracování různých událostí jako je kliknutí na tlačíko či změna položky ve výběrovém seznamu.

6.4.5 Komponenta NMFTopicDetection

Komponenta NMFTopicDetection stojí mimo komponentu DRMethods, protože neposkytuje přímo volání jedné z metod redukcí. Tato komponenta a její třída *TopicDetection* poskytuje přístup již ke konkrétní aplikaci metody NMF, k detekci témat.

NMFTopicDetection poskytuje dvě veřejné metody *FitDocumentsToTopics()* a *PrintTopics()* a jednu privátní metodu *SortAndPrint(double[,] arrayToSort, int index, int numberOfTerms)*. Metoda *PrintTopics* vrátí seznam témat na základě parametrů (počet témat a počet slov na téma), tato témata získá pomocí metody *SortAndPrint*. Pomocí metody *FitDocumentsToTopics* zařadí jednotlivé dokumenty do témat a vrátí textovou reprezentaci tohoto zařazení.

Princip detekce témat popisuje experiment v kapitole 8.3.

6.4.6 Komponenta TFIDFExample

Jako samotná komponenta vystupuje v projektu TFIDFExample, která poskytuje metody pro vytvoření TF-IDF matice. Tato komponenta odpovídá projektu zmíněném v kapitole 6.3.2.

7 Experimenty nad obrázky

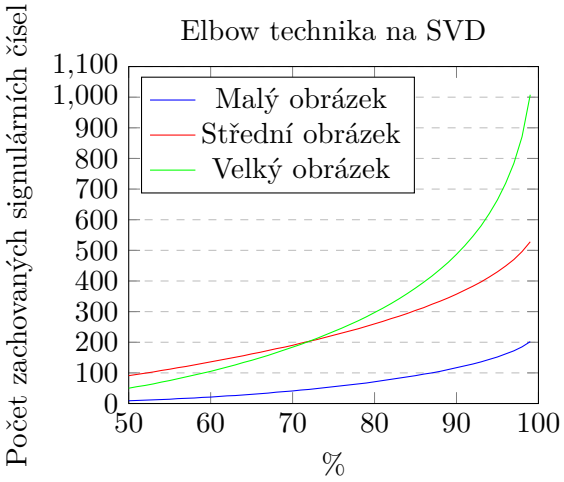
7.1 Techniky pro určení vhodného podprostoru

Otázkou pro tento experiment je, kolik by mělo být zachováno vlastních čísel, aby byl výsledek redukce kvalitní.

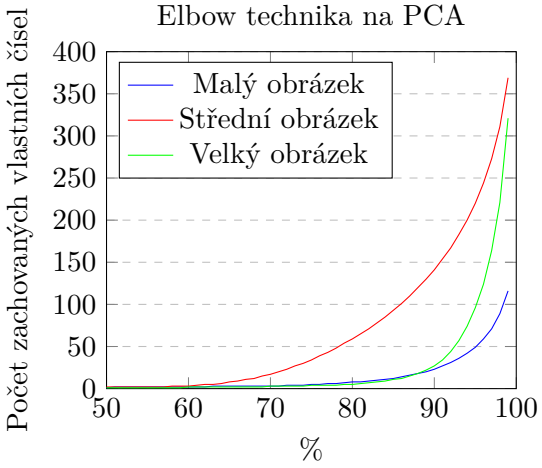
Tento experiment bude prováděn na obrázcích z toho důvodu, že je u nich jednoduché určit, kdy už je množství zachovaných vlastních čísel optimální či ne. Porovnat obrázky můžeme totiž pouhým okem a na základě velikosti. Z důvodu rychlejšího výpočtu budou použity obrázky ve stupních šedi.

V případě použití metody PCA(KPCA)/SVD má každý obrázek reprezentovaný maticí určitý počet vlastních/singulárních čísel. Počet vlastních/singulárních čísel je určen šířkou obrázku. Např. malý obrázek je široký 350 pixelů a proto je počet vlastních/singulárních čísel 350. Pro střední obrázek je tento počet 580 a pro velký 2024 vlastních/singulárních čísel.

U SVD je možné použít obě techniky zmíněné v podkapitole 5.2.1. Nejprve budou shrnuty výsledky pro elbow techniku. Elbow technika u SVD netvoří vždy tzv. „lokýtek“. Tento lokýtek byl vytvořen pouze u velkého obrázku, viz. obrázek 9. U dalších byl tento pokles téměř lineární. Pro malé obrázky tvoří hranici pro optimální kompresi poměr 70%, tedy zhruba 35 singulárních čísel. U středního obrázku lze pomyslnou hranici posunout na 60% (asi 136 singulárních čísel). Podobná situace pak nastává i u velkého obrázku. I zde stačí zachovat poměr na 60%, z toho vyplývá, že se zachová asi 105 singulárních čísel.



Obrázek 9: Elbow technika u metody SVD



Obrázek 10: Elbow technika u metody PCA

U SVD se velmi osvědčila technika založená na průměru, kdy u malého obrázku bylo potřeba ponechat 35 singulárních čísel, přesně 70% jak u elbow techniky. U středních a velkých obrázků jsou však počty ponechaných vlastních čísel nad zadanou hranicí. U středních obrázků to bylo 177 vlastních čísel (asi 67% u elbow techniky) a u velkých 346 (asi 84%). Výsledky obou technik lze vidět na obrázku 11.

(a) Původní obrázek (b) Elbow technika s 80% ($r = 65$) (c) Technika založená na průměru ($r = 35$)

Obrázek 11: Použití elbow techniky a techniky založené na průměru u metody SVD

U PCA byly výsledky odlišné. Zatímco u SVD se „lokýtek“ tvořil jen u velkého obrázku, u PCA se tvořil vždy, viz. obrázek 10. Pro zvolený střední obrázek stačí usilovat o zachování minimálně 90% variability. Tedy zachovat 30% vlastních čísel. U malého obrázku je třeba být přísnější a zachovat minimálně 97% variability, což odpovídá 14,5% zachovaných vlastních čísel. U velkého obrázku je potřeba zachovat minimálně 96% variability, tedy asi 8% všech vlastních čísel.

Technika založená na průměru se u PCA oproti SVD neosvědčila. Dle této techniky by stačilo u malého obrázku ponechat 13 vlastních čísel (93% variability), což je o 5% méně než výše stanovené minimum. U středních obrázků je třeba ponechat asi 84 % variability, tedy 86 vlastních čísel. I zde se tato hodnota liší asi o 6% než je stanovené minimum pro přijatelný výsledek. Ani u velkého obrázku, kde podle techniky založené na průměru stačilo ponechat 63 vlastních čísel, tedy asi 93% variability. Zde však také minimální hranice nebyla splněna. Výsledky obou technik použitých nad metodou PCA s lze prohlédnout na obrázku 12.

(a) Původní obrázek (b) Elbow technika s 80% ($k = 50$) (c) Technika založená na průměru ($k = 13$)

Obrázek 12: Použití elbow techniky a techniky založené na průměru u metody PCA

Elbow technika funguje i pro KPCA, viz. experiment 7.2.

Vizuální porovnání obrázků může být velmi subjektivní, a proto se obrázky následně porovnávaly i podle velikostí, kde se výsledky SVD a PCA liší řádově v jednotkách kB.

Z experimentu bylo zjištěno, že pro určování vhodného podprostoru u obrázků se pro SVD více hodí technika založená na průměru, kdežto u PCA je vhodná naopak elbow technika.

Výsledky experimentu - obrázky, tabulky a grafy - je možné vidět v příloze C. V příloze jsou zobrazeny výsledky pro středně velké obrázky. Další naměřené výsledky je možné nalézt na přiloženém CD.

7.2 Kernel PCA a jeho účinnost na obrázcích

Při testování toho, zda lze kernel PCA použít na obrázky podobně jako PCA, bylo třeba nalézt vhodné hodnoty parametrů pro jednotlivé kernel metody. Proto se dvanáct různých kernel metod rozdělilo do čtyř skupin:

1. Kernel metody s parametrem sigma
2. Kernel metody s parametrem c
3. Kernel metody s parametrem d
4. Kernel metody s parametrem c a d

Ve většině skupin se nachází minimálně tři kernel metody. V poslední skupině se jako jediný nachází polynomiální kernel.

Pro jednotlivé metody bylo testováno několik hodnot a pomocí elbow techniky pro určení optimálního podprostoru bylo zjišťováno, který parametr je vhodný. Při experimentu byl použit malý obrázek. Elbow technika byla nastavena na 98%. Pokud při 98% nevydala kernel metoda požadovaný výsledek, byla použita technika založená na průměru. Pokud ani při použití techniky založené na průměru nebyl výsledek optimální, byl kernel považován za nepoužitelný na obrázcích. Nepoužitelný v tom smyslu, že nedokázal zachovat 98% variability nebo že výsledný obrázek nebyl ani zdaleka podobný tomu původnímu.

U metod s parametrem σ se volba parametru prováděla následovně. Exponenciálnímu a Cauchyovu kernelu byl přiřazen parametr σ v intervalu $< 0; 10^3 >$. U Laplaceova kernelu byl interval stanoven na $< 0; 10^4 >$. Pro gaussovský kernel byla situace jiná. Ten totiž v případě obrázků pracuje s velmi malými hodnotami parametru σ, a proto byla σ volena v intervalu $< 10^{-7}; 0, 1)$.

Metody s parametrem d byly testovány na hodnoty z intervalu <1;25>. Pokud však ani s jedním parametrem z intervalu nedávala metoda optimální výsledek, bylo otestováno ještě několik hodnot mimo tento interval.

U metod, kde jako parametr vystupuje konstanta c, byla nejprve provedena volba konstanty po desítkách. Při nedosažení optimálního výsledku se poté volila konstanta po jednotkách, případně stovkách. Někdy bylo nutné volit konstantu v desítkách či stovkách tisíc.

Polynomiální kernel tvořící samostatnou skupinu byl nejprve otestován na parametr d a při nalezení příznivého výsledku se otestovala konstanta c v intervalu $< 0; 10^4 >$.

7.2.1 Zjištěné výsledky

Pokud není řečeno jinak, používají kernel metody elbow techniku s 98% pro určení optimálního podprostoru.

Exponenciální kernel
Volba parametru σ by měla být v intervalu <10;70>. Obecně platí, že čím vyšší σ je, tím je komprese vyšší. Při použití centrování byl výsledek méně kvalitní.

Cauchyho kernel
U cauchy kernelu není dle výsledků dobré centrovat kernel matici. Parametr σ zde má velký rozsah, minimálně pokrývá celý testovaný interval $< 0; 10^3 >$. Mohl by být tedy volen i větší s tím, že komprese by byla silnější.

Laplaceův kernel
Laplaceův kernel je taktéž vhodný pro obrázky. Stejně jako cauchyho kernel, ani laplaceův by se neměl centrovat. Bylo zjištěno, že je možné volit parametr σ v intervalu $< 20; 10^4 >$ s tím, že může být volen i větší něž je stanovený interval.

Gaussovský kernel
U gaussovského kernelu je situace jiná než u třech předchozích. Parametr σ je ideální volit v intervalu $< 10^{-5}; 10^{-4} >$. Centrovat gaussovský kernel je možné, avšak výsledky nejsou optimální a je potřeba volit menší parametr σ.

Mocninový (power) kernel
U mocninového kernelu se projevila lépe technika založená na průměru, kdy kernel s touto technikou vydal výsledek pro d v intervalu <1;12>, avšak tento nebyl optimální. Centrování kernel matice výsledek zhoršilo.

Logaritmický kernel
I zde byla úspěšnější technika založená na průměru. Centrovat kernel matici za použití této techniky se nevyplatí. Naopak při použití elbow techniky je centrování velmi nápomocné a vede k optimálnímu výsledku, podobně jako technika založená na průměru bez centrování. Parametr d byl volen jako násobky 5 v intervalu <5; 25>. Bylo zjištěno, že pokud byl parametr nastaven na hodnotu 5, vydával stejné výsledky, jako by byl nastaven na hodnotu 25.

Zobecněný T-studentův kernel
Necentrovaný kernel vydá dobrý výsledek, ale komprese je velmi malá (asi 75% původní velikosti). V následujícím experimentu bude však požadována velikost 14,6 kB, které zobecněný

T-studentův kernel během měření nedosáhl. Centrování je možné použít, zhoršuje však kvalitu. Pro necentrovaný kernel je třeba volit d v intervalu <1;5>.

Kernel založený na hyperbolickém tangentu
Tento kernel byl shledán nepoužitelným pro aplikaci nad obrázky.

Multikvadratický kernel
Multikvadratický kernel komprimuje obrázky dvěma způsoby. Konstantu c lze volit v celém testovaném intervalu $< 1, 10^6 >$. Dokud je c menší než 260, tak obrázek vždy vypadá velmi podobně, pouze se mění „šmouhy“ způsobené kompresí a velikost výsledných obrázků se liší na základě toho, jak byly skvrny umístěny. Velikost komprese může být dotačující, ale kvalita ne. Pokud je konstanta větší nastává situace jako u zobecněného T-studentova kernelu. Komprese je asi 75% původní velikosti, což je pro následující experiment nedostatečné. Důležité však je, aby byla kernel matice nejprve vycentrována. Bez centrování jsou výsledky nepoužitelné.

Inverzní multikvadratický kernel
Bylo zjištěno, že inverzní multikvadratický kernel funguje s konstantou $c = a * 10$, kde a je libovolné celé číslo. Platí, že čím vyšší toto číslo je, tím je komprese větší. Centrovat kernel matici se nevyplatí, metoda bez centrování vydává solidní výsledky.

Racionální kvadratický kernel
U racionálního kvadratického kernelu byla situace s volbou konstanty podobná jako u multikvadratického kernelu, avšak bylo zapotřebí zvolit větší násobek. Bylo zjištěno, že pro nejlepší výsledek u racionálního kvadratického kernelu je třeba nastavit $c = a * 10^5$, kde $a \in$ <1,15>. Ani u tohoto kernelu se centrování nevyplatí.

Polynomiální kernel
Vhodným parametrem d je u obrázků hodnota 1 s tím, že kernel matice bude centrována. Po otestování volby konstanty c v intervalu $< 0; 10^4 >$ bylo ukázáno, že hodnota konstanty nemá vliv na výsledek výpočtu. Pro $d = 1$ však výsledek nebyl optimální.

Celkový výsledek tohoto experimentu poskytuje tabulka 14. Vyplývá, že šest kernel metod z dvanácti lze určitě použít pro kompresi obrázků, neboť jejich výsledky byly srovnatelné s metodou PCA. Multikvadratický kernel lze použít pro kompresi pouze do určité velikosti. Zkomprimuje obrázek asi na 75% jeho původní velikosti. Zobecněný T-studentův kernel se v případě necentrované matice chová stejně jako multikvadratický. Další tři kernel metod (Polynomiální, Logaritmická a Mocninová) nelze použít pro kompresi obrázku, protože snižují jeho kvalitu a kernel metodu založenou na hyperbolickém tangentu nelze použít vůbec, protože pro obrázky nevydala žádný optimální výsledek.

Pro ověření výsledků bylo provedeno měření i na středně velkém obrázku, kde lze tyto závěry s jistou tolerancí aplikovat stejně.

Účelem experimentu bylo ověření toho, že některé kernel metody jsou použitelné pro kompresi obrázků. Zda je tato komprese účinná či ne se zabývá experiment 7.3.

Výsledky experimentu si lze prohlédnout v příloze D a na přiloženém CD.

7.3 Porovnání výsledků jednotlivých metod

Tento experiment porovnává vybrané metody pro použití na kompresi obrázků. Pro kompresi byl použit obrázek spadající do kategorie malých obrázků. Rozměry obrázků jsou 350×240 pixelů, velikost 49,2 kB a po převedení na stupně šedi 24 kB.

7.3.1 Postup experimentu

PCA

Velikost výsledného obrázku po kompresi pomocí metody PCA bude použit jako požadovaný výsledek i pro ostatní metody. To znamená, že bude hledán obrázek se stejnou nebo velmi podobnou velikostí. V experimentu 7.1 byla určená spodní hranice pro optimální výsledky 97%, v tomto experimentu bude však procento zachované variability vyšší, a to 98%. Při tomto procentu zachované variability je totiž výsledný obrázek poměrně kvalitní.

Nejprve bylo potřeba vytvořit menší experiment. U PCA lze mimo kovarianční matici použít i korelační matici. Otázkou je, který typ matice je lepší pro výpočet PCA. Z výsledků vyplynulo, že pro kvalitnější kompresi je lepší použít matici kovarianční. Experiment je shrnut v příloze E.1.

SVD

U SVD není třeba vytvářet žádné další experimenty, protože vše potřebné bylo určeno v experimentu 7.1. Díky zmíněnému experimentu bylo zjištěno, že spodní hranice pro optimální výsledky u malého obrázku je 70%, proto se bude hledat odpovídající obrázek od této hranice nahoru. Případně lze použít i techniku založenou na průměru.

KPCA

Hledání odpovídajícího obrázku bude vyplývat z experimentu 7.2, kde bylo zjištěno, že z dvanácti vybraných kernel metod lze pro kompresi využít pouze polovinu z nich. Účelem tohoto experimentu bude zjistit, zda komprese pomocí KPCA vytvoří požadovaný obrázek či ne.

NMF

U metody NMF bylo nutné nejprve určit aproximační parametr r a následně prozkoumat, zda má toleranční parametr ϵ vliv na kvalitu obrázku. Vhodný aproximační parametr byl nalezen v intervalu <70;100> při použití $\epsilon = 1000$. Vzhledem k tomu, že NMF nastavuje počáteční hodnoty matic W a H náhodně, nelze přímo určit optimální aproximační parametr. V tomto experimentu bude použito $r = 85$. Pro nalezení vhodných matic pro výpočet byla několikrát spuštěna metoda NMF s $r = 85$ a poté nalezen obrázek s odpovídající velikostí jako ten, který

byl vytvořen metodou PCA. Jakmile byl obrázek nalezen, byly pro výpočet použity prvky matic uložené v textovém souboru, který odpovídal nalezenému obrázku.

Po nalezení vhodného aproximačního parametru, bylo zjišťováno to, zda má parametr ϵ důležitý vliv na kvalitu obrázku a také to, v jakém rozsahu ho volit. Pro tento experiment bylo voleno ϵ v intervalu $< 10^{-9}; 10^{8} >$. Z tohoto experimentu bylo zjištěno, že ϵ v intervalu $< 10^{6}, 10^{8} >$ zhoršuje kvalitu. Dále to, že velmi malé ϵ ($\epsilon \leq 1$) způsobuje, že metoda dlouho iteratuje. I přes velký počet iterací nejsou výsledky kvalitnější. Vhodný interval na testovaném obrázku byl stanoven na $< 100; 10^{4} >$. Kvalita se v tomto intervalu však zásadně nemění, a proto volba parametru ϵ nemá velký vliv na kvalitu komprimovaného obrázku. V experimentu bylo zvoleno $\epsilon = 1000$.

Krátce se o těchto experimentech zmiňují přílohy E.3 a E.4.

CUR

Pro CUR bylo třeba určit optimální procento zachovaných sloupců a řádků pro každoý algoritmus ze tří implementovaných. Pro toto určení byly naměřeny všechny kombinace zachovaných procent sloupců a řádků a po tomto měření byly výsledky analýzovány. Z analýzy vyplynuly tyto závěry:

1. Pro náhodný algoritmus je vhodné zvolit součet procent zachovaných řádků a procent zachovaných sloupců roven šedesátipěti s tím, že procento zachovaných řádků bude větší než 15%.

2. U CUR-L2 bylo zjištěno to, že součet by měl být minimálně sto. Platí zde že čím více sloupců/řádků ponecháme, tím méně zachovaných řádků/sloupců potřebujeme, např. 30%.

3. Algoritmus hrubé síly vydává výsledky, které jsou alespoň trochu podobné požadovanému obrázku až při součtu 130, přesto jeho výsledky nejsou moc kvalitní. Ty začínají být kvalitní při vysokých procentech zachovaných řádků a sloupců.

Více informací a obrázky je možné nalézt v příloze E.2.

Díky výše zmíněným závěrům byly následně vybrány obrázky, které velikostně odpovídály požadovanému obrázku.

Po nalezení odpovídajících obrázků velikosti 14,6 kB u všech metod, bylo nejprve provedeno porovnání na základě rychlosti algoritmů a následně vizuální porovnání.

Pro vizuální porovnání dobře posloužila kompozice obrázku.

Na obrázku se nachází čtyři objekty, díky kterým je možné porovnávat kvalitu komprese. Čím zřetelnější objekty jsou, tím je kvalita lepší. Prvním z objektů je vůz na seno (včetně koňů a vozky), dále dům, pes u řeky a žena, která pere u řeky prádlo. Objekty jsou vyznačeny na obrázku 13. Po zjištění viditelnosti objektů, byly výsledky porovnávány dle kvality komprese. Zde hrál velkou roli šum, rozostření a různé kazy. Je však třeba předpokládat, že velmi malý šum či rozostření se při kompresi může objevit, a proto budou tyto vady v jisté míře tolerovány.

Obrázek 13: Původní obrázek s označenými objekty

7.3.2 Výsledky experimentu

Výsledky rychlostí algoritmů lze rozdělit do třech skupin - běh do dvou sekund, běh do deseti sekund a běh nad deset sekund. Výsledné rychlosti si lze prohlédnout v tabulce 15. Z důvodu mnoha prováděných iterací metodou NMF pomocí násobícího pravidla, běžela tato metoda velmi dlouho, aby vydala kvalitní obrázek o velikost 14,6 kB. Průměrná délka běhu byla šest minut a patnáct vteřin. Časy běhů algoritmů shrnuje tabulka 15. Velká rychlost metody SVD je nejspíše zapříčiněna použitím knihovny namísto vlastní implementace.

Z následující sekvence obrázků 14 lze vypozorovat, že tři ze čtyř objektů jsou viditelné velmi dobře. U čtvrtého objektu, ženy, zůstala viditelná spodní část šatů a žena se proto jeví jen jako tmavý bod v obrázku. Šum a kazy u jednotlivých metod nejsou moc velké. Problémem může být menší rozostření u metody NMF.

(a) PCA ($k = 50$) (b) SVD ($r = 53$) (c) NMF ($r = 85$)

Obrázek 14: Komprese obrázků pomocí metod PCA, SVD a NMF

U metody CUR (obrázek 15) a jejích jednotlivých algoritmů je situace jiná.

Nejlépe dopadl algoritmus náhodného výběru, který vydává stejné výsledky jako předchozí algoritmy. Jeho nevýhodou může být poměrně velký šum. Algoritmus hrubé síly a CUR-L2 algoritmus nevydaly pro požadovanou velikost optimální výsledky. Vydané výsledky obsahují

(a) Náhodný CUR (26% řádků a 80% sloupců) (b) Bruteforce CUR (21% řádků a 89% sloupců) (c) CUR-L2 (13% řádků a 88% sloupců)

Obrázek 15: Komprese obrázků pomocí metody CUR a jejich algoritmů

hodně kazů a taktéž nezachovávají všechny objekty. Vzhledem k tomu, že je náhodný algoritmus CUR založen na náhodě, mohou se výsledky lišit. Neměly by se však pro zvolené procento zachovaných sloupců a řádků lišit o mnoho.

Výsledky kernel metod, které jsou zobrazeny na obrázku 16 vypovídají o tom, že vyvozené závěry pro PCA platí i pro KPCA. Výsledné obrázky jednotlivých kernel metod jsou srovnatelné s výsledkem metody PCA.

(a) Gaussovské KPCA ($\sigma = 6.8^{-7}$) (b) Cauchyho KPCA ($\sigma = 1225$) (c) Komprese pomocí Laplaceova KPCA ($\sigma = 7300$)

(d) Exponenciální KPCA ($\sigma = 6.8^{-7}$) (e) Racionálně kvadratické KPCA ($c = 1500000$) (f) Inverzní multikvadratické KPCA ($c = 850$

Obrázek 16: Komprese obrázků pomocí různých KPCA

Na základě rychlosti a kvality komprese bylo z dvanácti metod vybráno pět nejlepších v tomto pořadí:

1. SVD
2. PCA

3. Náhodný CUR

4. Inverzní multikvadratické KPCA

5. Racionální kvadratické KPCA

SVD a PCA byly vybrány pro velkou rychlost algoritmu a dobrou kvalitu komprese. Náhodný CUR byl umístěn na třetí místo z důvodu rychlosti. Na druhou stranu ale musíme brát v potaz vyšší míru šumu. V dnešní době však hraje každá vteřina navíc velkou roli, a proto byla rychlost před touto vadou upřednostněna. Čtvrté a paté místo patří kernel metodám. Horší umístění bylo způsobeno delší dobou běhu algoritmů. Výhodou je kvalita komprese srovnatelná s PCA. Pokud by výše zmíněné pořadí pokračovalo, následovaly by zbylé kernel metody seřazeny sestupně dle doby běhu algoritmu. Poslední tři místa by patřily metodě NMF, CUR algoritmu pomocí hrubé síly a CUR-L2 algoritmu.

8 Použití vybraných metod nad dokumenty

Pro většinu aplikací nad dokumenty bude pro snadnou demonstraci použita následující matice termů-dokumentů znázoněná v tabulce 5.

Tabulka 5: Matice termů-dokumentů pro aplikace

Popisky	**D1**	**D2**	**D3**	**D4**	**D5**	**D6**
bank	0	3	0	0	2	0
money	0	1	0	0	0	1
finance	0	2	0	0	3	0
sport	1	0	0	0	0	2
club	1	0	0	0	0	2
football	2	0	0	0	0	2
show	0	0	1	2	0	0
actor	0	0	1	3	0	0
film	0	0	2	0	0	0

8.1 Latentní sémantické indexování (LSI)

Latentní sémantické indexování se používá pro ohodnocení sady dokumentů na základě dotazu a jejich relevantnost k tomuto dotazu. Pro výpočet se využívá metody SVD a nízkoúrovňové aproximace. Postup je následující:

1. **Vytvoř dotaz *q* na základě zadaných slov**
   Dotaz bude reprezentován vektorem velikosti m, kde m je počet slov (termů). V tomto vektoru mohou být pouze dvě hodnoty prvků - 0 nebo 1, kde 1 znamená, že je slovo do dotazu začleněno a 0 znamená opak.

2. **Na vstupní matici dokumentů-termů použij singulární rozklad**
   Vydá jako výstup matice U, Σ a V.

3. **Použij nízkoúrovňovou aproximaci do r-dimenzionálního prostoru, kde** $r \ll n$
   Maticím z předchozího kroku redukuje počet sloupců na r a vydá matice U_r, Σ_r a V_r. Na závěr matici V_r transponuje.

4. **Najdi nové vektory pro dokumenty**
   Nové vektory pro dokumenty jsou uloženy ve sloupcích matice V_r^T.

5. **Najdi nový vektor pro dotaz reprezentovaný v r-dimenzionálním prostoru**
   Nový vektor získáme provedením rovnice 17.

$$q = q^T U_k \Sigma_k^{-1} \tag{17}$$

   Transponovaný vektor pro dotaz se vynásobí s maticí U_k a inverzní maticí Σ_k.

6. **Ohodnoť dokumenty pomocí cosinovy míry podobnosti pro dotaz-dokument**
   Podobnost pomocí cosinovy míry lze vypočítát z rovnice 18.

$$cossim(q, d) = \frac{q \cdot d}{|q||d|} \tag{18}$$

Pro příklad poslouží matice obsahující náhodně vygenerované věty ekonomických článků 6. Jako dotaz bude položena sekvence slov *growth, price* a *oil*. Dotaz byl konstruován tak, aby každý dokument neobsahoval alespoň jedno slovo z dotazu.

Tabulka 6: Matice termů-dokumentů pro demonstraci LSI

Popisky	**D1**	**D2**	**D3**
world	1	1	1
economy	1	0	0
price	1	0	1
growth	0	2	0
gas	1	1	0
oil	0	1	1
bank	0	0	1

Aby nebylo zabráno místo výslednými maticemi po rozkladu SVD a výslednými maticemi po nízkoúrovňové aproximaci, bude příklad začínat až od bodu 4. Problém bude redukován na dvě dimenze. Vektor pro dotaz bude vypadat následovně.

$$q^T = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Vektory pro jednotlivé dokumenty vyšly následovně:

1. dokument - (0,4544; -0,5418)
2. dokument - (0,7662; 0,6426)
3. dokument - (0,4544; -0,5418)

Nový vektor pro dotaz byl (0,3422; 0,2327). Po aplikování rovnice 18 byla vypočtena cosinova podobnost mezi dotazem a dokumentem.

$$cossim(q, d1) = \frac{(0,3422)(0,2327) + (0,4544)(-0,5418)}{\sqrt{(0,3422)^2 + (0,2327)^2}\sqrt{(0,4544)^2 + (-0,5418)^2}} = 0,1005$$

$$cossim(q, d2) = \frac{(0,3422)(0,2327) + (0,7662)(0,6426)}{\sqrt{(0,3422)^2 + (0,2327)^2}\sqrt{(0,7662)^2 + (0,6426)^2}} = 0,9949$$

$$cossim(q, d3) = \frac{(0,3422)(0,2327) + (0,4544)(-0,5418)}{\sqrt{(0,3422)^2 + (0,2327)^2}\sqrt{(0,4544)^2 + (-0,5418)^2}} = 0,1005$$

Z podobností vyplývá, že nejlepší výsledek pro hledaný dotaz je druhý dokument, který obsahuje slovo *growth* ve větší míře než ostatní dokumenty. Další dva dokumenty mají stejnou podobnost s dotazem. Je to způsobeno tím, že mají stejný vektor. Oba obsahují čtyři slova, z toho dvě stejná. Na tomto příkladu lze vypozorovat, že i když třetí dokument obsahuje více slov dotazu než první, je jejich podobnost s dotazem stejná.

8.2 Vizuální reprezentace dokumentů pomocí PCA a KPCA

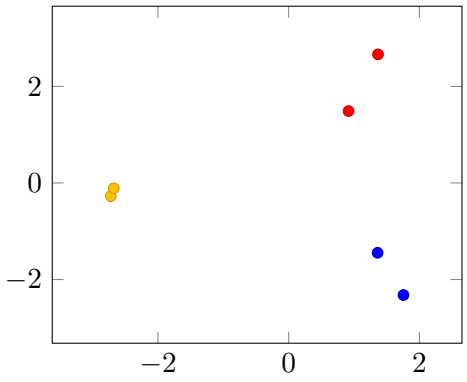
Vizuální reprezentace dokumentů pomocí PCA a KPCA lze realizovat tak, že nejprve spustíme algoritmus 1 pro PCA nebo algoritmus 2 pro KPCA. Tento algoritmus na základě vstupní matice o rozměrech $m \times n$ vydá matici $m \times k$, kde k je počet dimenzí.

Výsledky matice lze poté předat jako souřadnice grafu libovolnému programu, který umí grafy vykreslovat.

Pro příklad bude použita matice v tabulce 5. Dokumenty budou vizualizovány do dvoudimenzionálního prostoru. PCA na základě této matice vydá matici zaznamenanou v tabulce 7.

Tabulka 7: Výsledek metody PCA při redukci do 2D

Dokument	X	Y
1	1,3605	-1,4455
2	-2,7220	-0,2738
3	0,9156	1,4881
4	1,3666	2,6626
5	-2,6764	-0,1103
6	1,7557	-2,3211



Obrázek 17: Vizualizace dokumentů do 2D

Z vizualizace na obrázku 17 lze vypozorovat, že jsou v kolekci šesti dokumentů tvořeny tři skupiny. Tyto skupiny by se mohly chápat jako kategorie či témata, které lze v kolekci definovat. S detekováním tématu z kolekce dokumentů může pomoci metoda NMF.

Tyto výsledky vydávaly i všechny vybrané kernel metody.

8.3 Detekce témat pomocí metody NMF

Za pomocí NMF lze detekovat témata pro dokumenty z určité kolekce. Počet témat určuje aproximační parametr r a díky němu metoda NMF rozloží matici A na dvě matice W $(m \times r)$ a H $(r \times n)$. V případě detekce témat není potřeba pro získání výsledku matice zpětně násobit, neboť jsou všechny informace uloženy v těchto dvou maticích.

V matici W jsou uloženy hodnoty k určení toho, jak moc daný dokument spadá do daného tématu. Platí, že čím vyšší číslo je, tím je vazba na téma silnější. Podobná situace nastává i u matice H. V matici H jsou uloženy hodnoty, které vyjadřují sílu slova (termu) v daném tématu. Pro určení tématu se vybere k prvních slov, tedy k slov, které mají v určitém řádku matice nejvyšší hodnoty.

Pro demonstrativní příklad bude použita matice zaznamenaná v tabulce 5. Metoda NMF rozloží tuto matici na dvě následující.

$$W = \begin{pmatrix} 0 & 2,961 & 0 \\ 1,815 & 0,122 & 0 \\ 0 & 0 & 1,577E^{-30} \\ 0 & 0 & 1.3E^{9} \\ 1,743 & 2,492E^{-140} & 0 \\ 0,021 & 4,528 & 0 \end{pmatrix}$$

$$H = \begin{pmatrix} 1,41 & 0,282 & 1,399 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2,287E^{-5} & 0,156 & 5,646E^{-16} & 0,41 & 0,41 & 0,511 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1,538E^{-9} & 2,308^{-9} & 0 \end{pmatrix}$$

Z matice W lze vyčíst, že každý dokument spadá do určitého tématu. Díky matici H se mohou tématu přiřadit klíčová slova, pomocí nichž lze téma definovat. Téma může být určeno libovolným počtem slov, v tomto příkladě to budou slova dvě. Např. první řádek matice H vypovídá o tom, že téma lze definovat prvním a třetím slovem z matice termů-dokumentů. Z příkladu bylo zjištěno, že:

1. Do tématu č. 1 definovaného slovy *bank* a *finance* spadá 2. a 5. dokument.
2. Do tématu č. 2 definovaného slovy *football* a *sport* spadá 1. a 6. dokument.
3. Do tématu č. 3 definovaného slovy *actor* a *show* spadá 3. a 4. dokument.

9 Experimenty nad dokumenty

Pro tyto experimenty byla použita již dříve zmíněná kolekce článků z BBC [45]. V této kolekci se nachází 2225 článků v pěti kategoriích. Z ní byly vybrány čtyři kategorie (sport, ekonomika, politika a technika) a z každé kategorie bylo vybráno prvních 250 článků.

9.1 Detekce témat pomocí NMF

Experiment demonstruje použití detekce témat pomocí metody NMF na reálných datech. Bylo testováno několik témat s různým počtem dokumentů. Je třeba zmínit, že metoda dostupná v knihovně *Accord.NET* je napsána tak, že vydá výsledek po určitém počtu iterací a není založena na tom, že konverguje pomocí frobeniovy normy. Proto bylo naměřeno pro každou testovanou sadu padesát měření pro matici TF-IDF a padesát pro matici dokumentů-termů. Pro každou matici byla spuštěna metoda NMF pro sto až tisíc iterací. Kvůli náhodné inicializaci matic bylo provedeno pro každou iteraci vždy pět pokusů.

9.1.1 Dvě témata

Pro dvě témata byly použity kolekce pro téma fotbal a ekonomika. Kolekce obsahovala 250 článků pro každé téma. Tabulka 8 znázorňuje, jak byla detekce tématu úspěšná pro různý počet iterací a různé použité matice. Tedy kolik bylo úspěšných pokusů z pěti možných.

Tabulka 8: Detekce témat pro 500 článků ve dvou kategoriích

Typ matice / Počet iterací	100	200	300	400	500	600	700	800	900	1000
Term-dokument	0	0	0	0	0	0	0	0	0	0
TF-IDF	1	1	4	3	4	3	3	2	5	3

Z tabulky lze vyvodit, že matice termů-dokumentů nebyla pro toto měření vhodná. Na druhou stranu matice TF-IDF začala vydávat výsledky již po nastavení iterací na 300. Od tříset po tisíc iterací vydala správná témata průměrně ve třech pokusech z pěti.

9.1.2 Tři témata

Témata se detekovaly pro články v kategoriích sport, ekonomika a politika. Tabulka 9 demonstruje výsledky detekce témat pro 450 článků, tedy 150 z každé kategorie.

Tabulka 9: Detekce témat pro 450 článků ve třech kategoriích

Typ matice / Počet iterací	100	200	300	400	500	600	700	800	900	1000
Term-dokument	0	1	0	0	1	0	0	0	1	0
TF-IDF	5	5	4	3	5	4	4	4	4	4

Stejně jako v předchozí kolekci, ani zde matice termů-dokumentů nevydala téměř nikdy správný výsledek. Matice TF-IDF začala vydávat výsledky již od počátku měření. Průměrně detekovala témata článku správně ve čtyřech pokusech z pěti.

9.1.3 Čtyři témata

Pro čtyři témata bylo vybráno celkově 500 souborů ve čtyřech kategoriích - sport, ekonomika, politika a technika. V tabulce 10 lze vidět výsledky pro jednotlivé počty iterací a typy matic.

Tabulka 10: Detekce témat pro 500 článků ve čtyřech kategoriích

Typ matice / Počet iterací	100	200	300	400	500	600	700	800	900	1000
Term-dokument	0	0	0	0	0	0	0	0	0	0
TF-IDF	3	0	2	1	0	3	2	2	3	4

Ani u posledního měření nebyla matice termů-dokumentů úspěšná. Podobně je tomu u matice TF-IDF, kde byla průměrná úspěšnost dva pokusy z pěti. Avšak od pětiset iterací byla tato úspěšnost stejná jako u kolekce s dvěma tématy, tedy tři úspěšné pokusy z pěti.

Výsledky mohou být ovlivněny také tím, že čím více slov kolekce obsahuje, tím je pro metodu náročnější téma detekovat. Bylo by tedy vhodné kolekci slov redukovat tím, že by se rozšířil slovník stopslov a vyřadila se slova, která nejsou podstatná či přídavná jména.

Závěrem lze říci, že pro detekci témat je lepší použít matici TF-IDF. Průměrná úspěšnost ze všech kolekcí je tři úspěšné pokusy z celkových pěti provedených pokusů.

9.2 Vizuální reprezentace dokumentů pomocí PCA

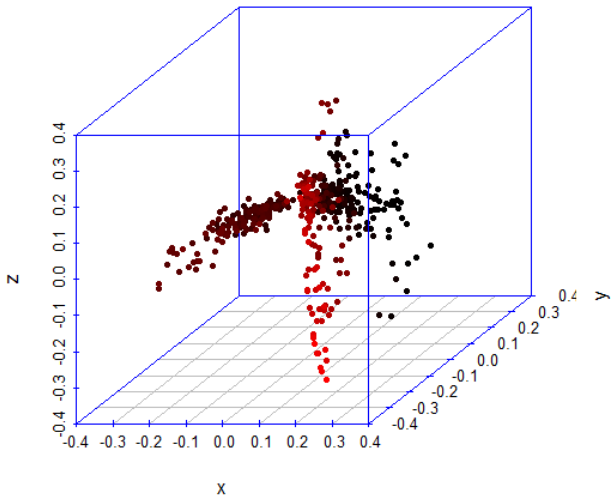
Účelem tohoto experimentu byla demonstrace vizuální reprezentace dokumentů pomocí metody PCA na větších datech a také zjistit to, zda je lepší pro reprezentaci použít matici termů-dokumentů či TF-IDF matici. Pro demonstraci byla zvolena kolekce 450 článků ve třech různých tématech. Vizuální reprezentace dokumentů ukazuje, jak moc si jsou dané dokumenty blízké.

9.2.1 Použití matice TF-IDF

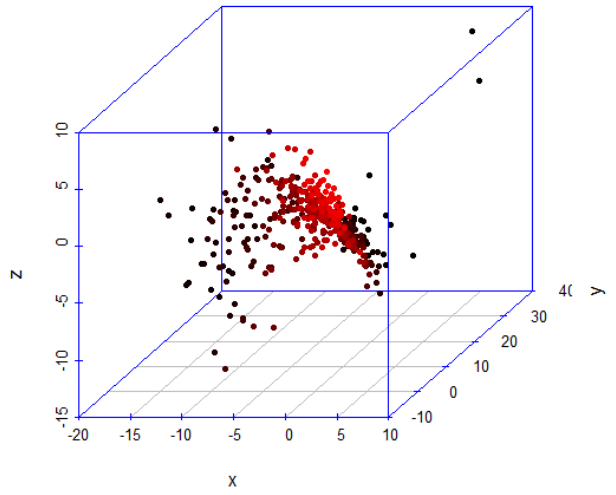
Na obrázku 18 lze vidět, že se po provedení metody PCA s maticí TF-IDF tvoří tři shluky dokumentů. Některé dokumenty se sice překrývají, což neznačí problém vizualizace, ale spíše to, že jsou dokumenty již na hranici shluku.

9.2.2 Použití matice termů-dokumentů

Výsledek vizualizace dokumentů pomocí metody PCA, která pro výpočet použila matici termů-dokumentů lze vidět na obrázku 19. Je možné vypozorovat, že většina dokumentů se shlukuje v jednom místě, a tím pádem oddělení dokumentů není až natolik přesné jako u matice TF-IDF.



Obrázek 18: Vizualizace dokumentů pomocí PCA - matice TF-IDF



Obrázek 19: Vizualizace dokumentů pomocí PCA - matice termů-dokumentů

Z výše uvedených obrázků lze vyvodit, že s použitím matice TF-IDF se dosáhne lepší vizualizace.

Závěrem lze říci, že je možné použít vizualizaci dokumentů pomocí metody PCA. Celkové výsledky vizualizace pomocí těchto dvou matic jsou ovlivněny na základě toho, jak matice vyhodnocují, které slovo je významější a které ne. U matice termů-dokumentů je významné to slovo, jehož četnost je v daném dokumentu nejvyšší. Naopak matice TF-IDF vyhodnocuje důležitost slova dle délky dokumentu a jeho četnosti. Platí, že čím vyšší četnost slova v dokumentu je, tím je méně důležité.

10 Závěr

Metody pro redukci dimenzionality jsou velmi nápomocné při zpracování a analýze velkých dat, které se v dnešní době prakticky využívají v mnoha odvětvích, jako je průmysl, zdravotnictví či sociální sítě. Jejich hlavním úkolem je redukovat dimenzi problému, čímž mohou poskytnout smysluplnější reprezentaci dat a snížít časovou náročnost výpočtu.

Cílem práce bylo implementovat a použít metody pro redukci dimenzionality na různých datových kolekcích. Bylo vybráno pět lineárních i nelineárních metod - SVD, PCA, CUR, NMF a KPCA. Jako kolekce byly zvoleny obrázky a dokumenty. U obrázků byla provedena komprese a u dokumentů se oblast použití lišila, dle toho, na co je metoda určená. U metody SVD to bylo latentní sémantické indexování, PCA a KPCA byly použity pro vizuální reprezentaci sady dokumentů a metoda NMF detekovala témata pro zvolenou kolekci dokumentů a následně řadila jednotlivé dokumenty do vytvořených témat.

Prvním experimentem byla komprese obrázků. Aplikace umožňuje pracovat jak s obrázky ve stupních šedi, tak s barevnými obrázky. Experimenty byly z důvodu rychlosti výpočtu prováděny na obrázcích ve stupních šedi. Bylo ukázáno, že všechny vybrané metody lze využít pro kompresi obrázků. Nejlépe dopadla metoda SVD, následována metodou PCA a na pomyslném třetím místě skončil náhodný algoritmus CUR. Pro tento experiment bylo potřeba provést několik dílčích experimentů pro zjištění optimálních parametrů. Jeden z experimentů zjišťoval, zda je možné na zvolených obrázcích použít různé kernel metody pro kernel PCA. Bylo ukázáno, že šest kernel metod z dvanácti je možné použít tak, že vydají stejný výsledek jako PCA. Dále byl u metod SVD a PCA proveden experiment s technikami určující vhodný podprostor, kdy se u PCA osvědčila tzv. „elbow“ technika a u SVD technika založená na průměru.

Druhý experiment byl prováděn nad kolekcí dokumentů. Nejprve bylo na jednoduchých příkladech demonstrováno použití metod v oblasti zpracování dokumentů. U metody SVD bylo provedeno latentní sémantické indexování, které je vhodné při vyhledávání v kolekci dokumentů na základě dotazu. Díky metodě NMF bylo možné detekovat témata ze sady dokumentů, které spadaly do různých kategorií. Metoda NMF také umožňuje zařadit dokumenty do jednotlivých témat. Metody PCA a KPCA reprezentovaly dokumenty do prostorů s nižší dimenzí. Pro metody NMF a PCA byly učiněny experimenty nad kolekcí článků. Experimenty ukázaly, že detekce témat pomocí NMF je poměrně úspěšná, i přes náhodnou inicializaci matic. Vizualizace dokumentů ukázala, že se vytváří jisté shluky dokumentů, ale pro větší přesnost je lepší spustit vizualizovat výsledek shlukování, např. pomocí algoritmu k-means. Provedené experimenty ukazují, že pro práci s dokumenty byla lepší matice TF-IDF.

Možným vylepšením této práce by byla určitě paralelizace s případným přepsáním programu do jazyka C++, což by přineslo menší časovou náročnost všech výpočtů. Pro reprezentaci dokumentů se používaly dva typy matic a dalším vylepšením by mohlo být navržení lepšího slovníku stopslov, například pomocí některé z knihovem na zpracování přirozeného jazyka, která umožňuje vyhodnocovat slova na základě jejich slovního druhu a bylo by tak možné ponechat jen

podstatná a přídavná jména. Dále by mohla být aplikace rozšířena o další metody redukce dimenzionality a více jejich použití.

Díky této práci jsem získal zkušenosti v oblasti redukce dimenzionality a uvědomil jsem si důležitost této problematiky v návaznosti na zpracování velkých dat. Dalším přínosem bylo obohacení znalostí v oblasti návrhu aplikace, která byla navržena tak, aby se některé třídy daly použít samostatně, případně za pomocí knihovních tříd.

Literatura

[1] VAN DER MAATEN, Laurens; POSTMA, Eric; VAN DEN HERIK, Jaap. Dimensionality reduction: a comparative. J Mach Learn Res, 2009, 10: 66-71.

[2] CÍCHA, Tomáš. *Řídké matice a jejich použití v numerické matematice* [online]. Brno, 2009 [cit. 2016-03-08]. Bakalářská práce. Masarykova univerzita, Přírodovědecká fakulta. Vedoucí práce Jiří Zelinka Dostupné z: `http://is.muni.cz/th/207863/prif_b/`.

[3] Ortogonální matice. Peter Franek [online]. Praha [cit. 2016-03-08]. Dostupné z: `http://www1.karlin.mff.cuni.cz/~pfranek/texty/unitmat.pdf`

[4] Metody řešení normálních rovnic. IngGeo - portál inženýrské geodézie [online]. [cit. 2016-03-08]. Dostupné z: `http://inggeo.fsv.cvut.cz/wiki/doku.php?id=04_teorie_chyb:0419_metody_reseni_normalnich_rovnic`

[5] Charakteristika variability [online]. [cit. 2016-03-08]. Dostupné z: `http://cit.vfu.cz/statpotr/POTR/Teorie/Predn1/variabil.htm`

[6] Pravděpodobnost a matematická analýza [online]. [cit. 2016-03-08]. Dostupné z: `http://euler.fd.cvut.cz/publikace/files/skripta3.pdf`

[7] Matematická statistika [online]. [cit. 2016-03-08]. Dostupné z: `http://www1.karlin.mff.cuni.cz/~hudecova/education/archive11/download/chem_predn/predn_slides_05.pdf>`

[8] SMITH, Lindsay I. A tutorial on principal components analysis, 2002, Dostupné z: `http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf`

[9] DOSTÁL, Zdeněk a Vít VONDRÁK. Lineární algebra [online]. Ostrava, 2011 [cit. 2016-03-08]. Dostupné z: `http://homel.vsb.cz/~jan939/LAIT/linearni_algebra.pdf`

[10] Metoda hlavních komponent [online]. [cit. 2016-03-08]. Dostupné z: `http://user.mendelu.cz/urban/doc/gacr/genstat-PCA-metoda.pdf`

[11] Metody pro redukci dimenze v mnohorozměrné statistice a jejich výpočet. [online]. 2014, 25(1) [cit. 2016-03-14]. ISSN 1210-8022. Dostupné z: `http://hdl.handle.net/11104/0226855`

[12] SORZANO, Carlos Oscar Sánchez; VARGAS, Javier; MONTANO, A. Pascual. A survey of dimensionality reduction techniques. arXiv preprint arXiv:1403.2877, 2014.

[13] SAUL, Lawrence K., et al. Spectral methods for dimensionality reduction. Semisupervised learning, 2006, 293-308.

[14] Linear and Quadratic Discriminant Analysis. Scikit-learn [online]. [cit. 2016-03-23]. Dostupné z: http://scikit-learn.org/stable/modules/lda_qda.html

[15] Linear Discriminant Analysis. Sebastianraschka [online]. 2014 [cit. 2016-03-14]. Dostupné z: http://sebastianraschka.com/Articles/2014_python_lda.html

[16] Multidimensional scaling [online]. [cit. 2016-03-14]. Dostupné z: http://forrest.psych.unc.edu/teaching/p208a/mds/mds.html

[17] WICKELMAIER, Florian. An Introduction to MDS Sound Quality Research Unit, Aalborg University, Denmark. 2003.

[18] TENENBAUM, Joshua B.; DE SILVA, Vin; LANGFORD, John C. A global geometric framework for nonlinear dimensionality reduction. science, 2000, 290.5500: 2319-2323.

[19] Mathematics of Data: Isomap and LLE [online]. [cit. 2016-03-14]. Dostupné z: http://www.math.pku.edu.cn/teachers/yaoy/Spring2011/lecture08.pdf

[20] SKILLICORN, David. Understanding complex datasets: data mining with matrix decompositions. CRC press, 2007.

[21] Applications of SVD: image compression [online]. 2011 [cit. 2015-11-14]. Dostupné z: https://inst.eecs.berkeley.edu/~ee127a/book/login/l_svd_apps_image.html

[22] Latent semantic indexing [online]. 2009 [cit. 2015-11-14]. Dostupné z: http://nlp.stanford.edu/IR-book/html/htmledition/latent-semantic-indexing-1.html

[23] Principal Component Analysis applied to digital image compression [online]. 2012 [cit. 2015-11-14]. Dostupné z: http://dx.doi.org/10.1590/S1679-45082012000200004

[24] RICHARDSON, Mark. Principal component analysis. Mathematical Modelling and Scientific Computing, University of Oxford, Oxford, UK, 2009.

[25] LA TORRE, Fernando De; NGUYEN, Minh Hoai. Parameterized kernel principal component analysis: Theory and applications to supervised and unsupervised image alignment. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE, 2008. p. 1-8.

[26] THURAU, Christian; KERSTING, Kristian; BAUCKHAGE, Christian. Deterministic CUR for Improved Large-Scale Data Analysis: An Empirical Study. In: SDM. 2012. s. 684-695.

[27] MITROVIC, Nikola, et al. CUR decomposition for compression and compressed sensing of large-scale traffic data. In: Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on. IEEE, 2013. p. 1475-1480.

[28] YANG, Jiyan, et al. Identifying important ions and positions in mass spectrometry imaging data using CUR matrix decompositions. Analytical chemistry, 2015, 87.9: 4658-4666.

[29] YUAN, Zhijian; YANG, Zhirong; OJA, Erkki. Projective nonnegative matrix factorization: Sparseness, orthogonality, and clustering. Neural Process. Lett, 2009. s. 11-13

[30] MAUTHNER, Thomas, et al. Efficient object detection using orthogonal NMF descriptor hierarchies. In: Pattern Recognition. Springer Berlin Heidelberg, 2010. p. 212-221.

[31] XU, Wei; LIU, Xin; GONG, Yihong. Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. ACM, 2003. p. 267-273.

[32] KLEEDORFER, Florian; KNEES, Peter; POHLE, Tim. Oh Oh Oh Whoah! Towards Automatic Topic Detection In Song Lyrics. In: ISMIR. 2008. p. 287-292.

[33] YANG, Jaewon; LESKOVEC, Jure. Overlapping community detection at scale: a nonnegative matrix factorization approach. In: Proceedings of the sixth ACM international conference on Web search and data mining. ACM, 2013. p. 587-596.

[34] CLINE, Alan Kaylor; DHILLON, Inderjit S. Computation of the singular value decomposition. Handbook of linear algebra, 2006, 45.1-45.13.

[35] MACIEJEWSKI, Anthony A.; KLEIN, Charles A. The singular value decomposition: Computation and applications to robotics. The International journal of robotics research, 1989, 8.6: 63-79.

[36] Implementing a Principal Component Analysis (PCA). Sebastianraschka [online]. [cit. 2016-03-23]. Dostupné z: `http://sebastianraschka.com/Articles/2014_pca_step_by_step.html`

[37] MASNAN, Maz Jamilah, et al. Principal Component Analysis–A Realization of Classification Success in Multi Sensor Data Fusion. PRINCIPAL COMPONENT ANALYSIS–ENGINEERING APPLICATIONS, 2012, 1.

[38] ABDI, Hervé; WILLIAMS, Lynne J. Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics, 2010, 2.4: 433-459.

[39] Kernel Methods for Pattern Analysis [online]. Cambridge: Cambridge University Press, 2004, 2007 [cit. 2015-11-05]. Dostupné z: `http://www.kernel-methods.net/kernels.html`

[40] WU, Yu-Chieh; YANG, Jie-Chi; LEE, Yue-Shi. An approximate approach for training polynomial kernel SVMs in linear time. In: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions. Association for Computational Linguistics, 2007. p. 65-68.

[41] MIKA, Sebastian, et al. Kernel PCA and De-Noising in Feature Spaces. In: NIPS. 1998. p. 7.

[42] WANG, Shusen; ZHANG, Zhihua. Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling. The Journal of Machine Learning Research, 2013, 14.1: 2729-2769.

[43] WANG, Shusen; ZHANG, Zhihua. A scalable cur matrix decomposition algorithm: Lower time complexity and tighter bound. In: Advances in Neural Information Processing Systems. 2012. p. 647-655.

[44] What does tf-idf mean? [online]. [cit. 2016-03-17]. Dostupné z: `http://www.tfidf.com/`

[45] BBC Datasets [online]. [cit. 2016-03-17]. Dostupné z: `http://mlg.ucd.ie/datasets/bbc.html`

[46] Accord.NET framework. Accord.NET framework [online]. [cit. 2016-03-23]. Dostupné z: `http://accord-framework.net/`

[47] TF*IDF in C# .NET for Machine Learning - Term Frequency Inverse Document Frequency. Primary Objects [online]. [cit. 2016-03-23]. Dostupné z: `http://www.primaryobjects.com/2013/09/13/tf-idf-in-c-net-for-machine-learning-term-frequency-inverse-document-frequency/`

A Obsah přiloženého CD

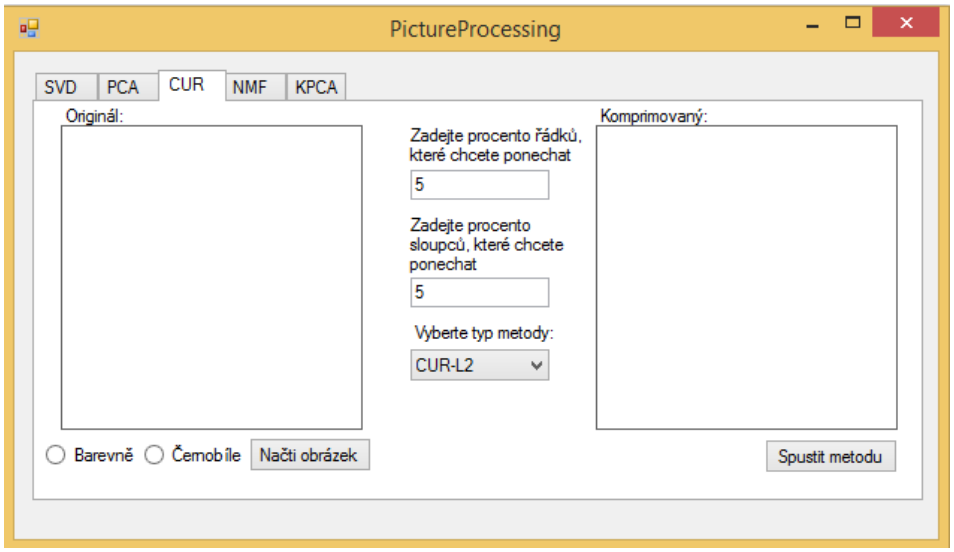
1. Testovací aplikace včetně zdrojových kódů
2. Obrázky
   (a) Testovací obrázky
   (b) Výsledné obrázky pro metodu SVD
   (c) Výsledné obrázky pro metodu PCA
   (d) Výsledné obrázky pro metodu KPCA
   (e) Výsledné obrázky pro metodu NMF
   (f) Výsledné obrázky pro metodu CUR
3. Dokumenty
   (a) Testovací kolekce dokumentů od BBC
   (b) Výsledky detekce tématu

Pro detailnější informace je v některých složkách vytvořen textový soubor s popisem složitější struktury složky.

B Návod k použití

B.1 Desktopová aplikace pro obrázky

Grafické rozhraní aplikace pro práci s obrázky je možné vidět na obrázku 20. Aplikace je organizována do záložek, kde každá záložka odpovídá jedné metodě redukce dimenzionality. Po vybrání záložky je nutné zvolit, zda chceme kompresi provádět nad barevným obrázkem nebo obrázkem ve stupních šedi. V případě barevného obrázku je potřeba ještě zaškrtnout/odškrtnout políčko, zda zachovávat alfa kanál či ne. Po kliknutí na tlačítko *Načti obrázek* se pomocí souborového dialogu vybere obrázek ke kompresi, který se po převedení na matici zobrazí v rámečku *Originál*. Další kroky se liší na základě použité metody.



Obrázek 20: Desktopová aplikace pro obrázky

SVD

U SVD jsou dvě možnosti výpočtu. Buďto si striktně zvolíme pomocí parametru r, kolik vlastních čísel chceme ponechat, nebo necháme aplikaci počet vlastních čísel zvolit samotnou. Automatická volba parametru r se provádí pomocí technik pro určení vhodného podprostoru, které jsou popsány v kapitole 5.2.1. V prvním případě stačí pouze vyplnit parametr r. Ve druhém je třeba zaškrtnout políčko *Určit r automaticky* a v případě elbow techniky zadat, kolik procent vlastností by mělo být pokryto. Pro ušetření výpočetního času lze zvolit, zda má aplikace znovu počítat singulární rozklad nebo použít dříve vypočtený.

PCA

U metody PCA je situace obdobná jako u SVD, akorát místo parametru r se volí parametr k. Nastavení techniky pro určení vhodného podprostoru zde zůstává. Navíc si lze zvolit, jestli má

PCA počítat s kovarianční (výchozí) nebo korelační maticí. I zde je možné výpočet algoritmu PCA neprovádět a ponechat poslední výsledek.

CUR

Metoda CUR se liší oproti dvěma předchozím tím, že se zde volí procento řádků a sloupců, které chceme zachovat. Po vyplnění těchto údajů je potřeba vybrat metodu pro výpočet CUR, na výběr jsou CUR-L2, Random a Bruteforce (výchozí).

NMF

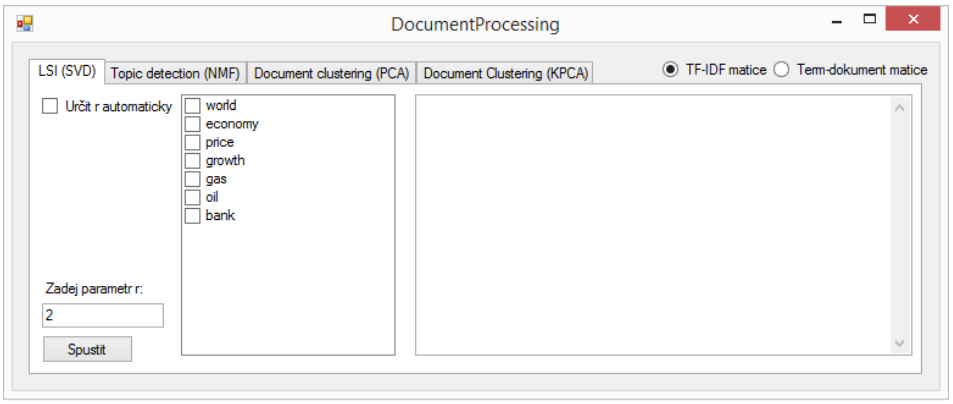
U metody NMF je potřeba nastavit aproximační parametr r, následně zvolit míru tolerance ϵ a zadat maximální počet iterací.

KPCA

Rozdíl mezi nastavením metody PCA a KPCA je v tom, že se mění pouze vypočtená matice. Zatímco u PCA se uživatel může rozhodovat mezi kovarianční a korelační maticí, u KPCA má na výběr z 12 typů kernel metod, které vypočtou kernel matici. Po výběru jedné z těchto metod musí uživatel zadat potřebné parametry, většinou jeden či dva. Na závěr uživatel, pomocí zaškrtávacího políčka, zvolí, zda chce nebo nechce kernel matici centrovat.

Po vyplnění parametrů stiskneme tlačítko *Spustit metodu*. Jakmile bude vše dokončeno, zobrazí se výsledný obrázek v rámečku *Komprimovaný*.

B.2 Desktopová aplikace pro dokumenty



Obrázek 21: Desktopová aplikace pro dokumenty

Stejně jako předchozí aplikace je i tato organizována do záložek, kde opět jedna záložka odpovídá jedné metodě, resp. možnosti použití této metody. Nejprve je třeba vybrat, zda se bude počítat s TF-IDF či maticí termů-dokumentů. Matice se počítá z dat uložených ve složce, kterou zvolíme. Volba souborů je možná pouze pro aplikace NMF a PCA. Aplikace SVD a KPCA jsou pouze demonstrativní, to znamená, že matice pro výpočet je pevně zadána. Následně zvolíme

jednu ze záložek. Další kroky se opět liší na základě použité metody. Obrázek 21 znázorňuje grafické uživatelské rozhraní aplikace pro dokumenty.

Latentní sémantické indexování (SVD)
U LSI lze zadat přesný počet zachovaných vlastních čísel nebo nechat program určit počet automaticky pomocí techniky pro určení vhodného podprostoru. Pokud je počet zachovaných vlastních čísel určen automaticky, je potřeba zvolit jakou technikou se bude výpočet provádět. Při použití elbow techniky je potřeba zadat, kolik procent vlastností je třeba zachovat. Pro výpočet LSI je nutné sestavit dotaz. Ten se sestaví pomocí seznamu zaškrtávacích políček. Po nastavení se stiskne tlačítko *Spustit* a výsledek je prezentován v pravé části okna.

Detekce témat (NMF)
U detekce témat (topic detection) je potřeba jako první načíst složku se soubory, ze kterých bude téma zjišťováno. Dále je potřeba zadat počet témat a slov, která budou toto téma charakterizovat. Jako poslední se nastaví počet iterací. Po stisknutí tlačítka *Spustit* se provede detekce témat nad vybranými dokumenty a výsledek je prezentován v pravé oblasti uživatelského rozhraní.

C Techniky pro optimální výběr vlastních čísel

Původní velikost obrázku byla 142 kB (barevný 163 kB).

C.1 Technika založená na průměru

Tabulka 11: Výsledky pro techniku založenou na průměru - střední obrázky

Metoda	Zachováno vlastních/singulárních čísel	Velikost	Zkomprimováno o
SVD	177	114 kB	28 kB
PCA	86	95,2 kB	46,8 kB

(a) Původní obrázek (b) SVD (c) PCA

Obrázek 22: Technika založená na průměru - střední obrázek

C.2 Elbow technika

(a) Původní obrázek (b) 90% (c) 95%

Obrázek 23: Použití elbow techniky u metody PCA

(a) Původní obrázek (b) 60% (c) 65%

Obrázek 24: Použití elbow techniky u metody SVD

Tabulka 12: Výsledky pro elbow techniku nad PCA - střední obrázky

%	λ	Velikost	Rozdíl
99	369	137 kB	4 kB
98	311	132 kB	9 kB
97	273	128 kB	13 kB
96	244	125 kB	16 kB
95	220	123 kB	18 kB
94	200	120 kB	21 kB
93	183	115 kB	26 kB
92	167	114 kB	27 kB
91	154	111 kB	30 kB
90	141	110 kB	31 kB
89	130	104 kB	37 kB
88	120	103 kB	38 kB
87	110	101 kB	40 kB
86	101	100 kB	41 kB
85	93	96,5 kB	44,5 kB
84	93	95 kB	46 kB
83	85	92,7 kB	48,3 kB
82	78	91,7 kB	49,3 kB
81	71	87,9 kB	53,1 kB
80	59	86 kB	55 kB
79	54	84,6 kB	56,4 kB
78	48	82,2 kB	58,8 kB
77	43	78,3 kB	62,7 kB
76	39	76,4 kB	64,6 kB
75	34	74,8 kB	66,2 kB
74	30	72,5 kB	68,5 kB
73	27	70,6 kB	70,4 kB
72	23	66,3 kB	74,7 kB
71	20	64,7 kB	76,3 kB
70	17	61,2 kB	79,8 kB
69	15	59,1 kB	81,9 kB
68	12	57,4 kB	83,6 kB
67	11	55,8 kB	85,2 kB
66	9	52,9 kB	88,1 kB
65	8	52 kB	89 kB
64	6	48,5 kB	92,5 kB
62 - 63	5	47,1 kB	93,9 kB
61	4	45,4 kB	95,6 kB
58 - 60	3	42,1 kB	98,9 kB
50 - 57	2	39,2 kB	101,8 kB

Tabulka 13: Výsledky pro elbow techniku nad SVD - střední obrázky

%	σ	Velikost	Rozdíl
98-99	496-528	141 kB	0 kB
95-97	430-470	140 kB	1 kB
94	413	139 kB	2 kB
92 - 93	383 - 397	138 kB	3 kB
91	370	137kB	4 kB
89 - 90	345 - 357	136 kB	5 kB
88	334	134 kB	7 kB
87	324	133 kB	8 kB
86	313	132 kB	9 kB
85	304	131 kB	10 kB
81 - 84	268 - 294	128 kB	13 kB
80	260	127 kB	14 kB
79	252	126 kB	15 kB
78	244	125 kB	16 kB
75 - 77	223 - 237	123 kB	18 kB
74	216	122 kB	19 kB
73	209	121 kB	20 kB
72	203	120 kB	21 kB
71	197	119 kB	22 kB
70	190	116 kB	25 kB
67 - 69	173 - 184	115 kB	26 kB
66	167	114 kB	27 kB
65	162	113 kB	28 kB
64	156	112 kB	29 kB
62 - 63	146 - 151	111 kB	30 kB
61	141	109 kB	32 kB
60	136	107 kB	34 kB
58 - 59	126 - 131	104 kB	37 kB
57	121	103 kB	38 kB
55 - 56	112 - 117	102 kB	39 kB
54	108	101 kB	40 kB
53	103	100 kB	41 kB
52	99	99,6 kB	41,4 kB
51	95	97,2 kB	43,8 kB
50	91	95,7 kB	45,3 kB

D Komprese obrázků pomocí kernel PCA

Pro srovnání je vždy jako první uveden výsledný obrázek metody PCA.

(a) PCA (14,6 kB) (b) $\sigma = 10$ (17,2 kB) (c) $\sigma = 60$ (17,2 kB)

Obrázek 25: Exponenciální KPCA s elbow technikou 98%

(a) PCA (14,6 kB) (b) $\sigma = 10$ (17,2 kB) (c) $\sigma = 1225$ (14,6 kB)

Obrázek 26: Cauchyho KPCA s elbow technikou 98%

(a) PCA (14,6 kB) (b) $\sigma = 20$ (17,3 kB) (c) $\sigma = 7400$ (14,6 kB)

Obrázek 27: Laplaceovo KPCA s elbow technikou 98%

(a) PCA (14,6 kB) (b) $\sigma = 1.0E - 5$ (17,3 kB) (c) $\sigma = 1.0E - 6$ (15,6 kB)

Obrázek 28: Gaussovské KPCA s elbow technikou 98%

(a) PCA (10,9 kB) (b) $d = 1$ (14,2 kB) (c) $d = 12$ (13,3 kB)

Obrázek 29: Mocninové KPCA s technikou založenou na průměru

(a) PCA (10,9 kB) (b) $d = 5$ (14,2 kB) (c) $d = 5$, centrovaná matice (11,5 kB)

Obrázek 30: Logaritmické KPCA s technikou založenou na průměru

(a) PCA (14,6 kB) (b) $d = 3$ (17,2 kB) (c) $d = 4$, centrovaná matice (14,2 kB)

Obrázek 31: T-Studentovo KPCA s elbow technikou 98%

(a) PCA (14,6 kB) (b) $c = 1$, centrovaná matice (14,9 kB) (c) $c = 20$, centrovaná matice (12,5 kB)

Obrázek 32: Multikvadratické KPCA s elbow technikou 98%

(a) PCA (14,6 kB) (b) $c = 100$ (17,2 kB) (c) $c = 800$ (14,9 kB)

Obrázek 33: Inverzní multikvadratické KPCA s elbow technikou 98%

(a) PCA (14,6 kB) (b) $c = 10^5$ (17,1 kB) (c) $c = 10^6$ (15,6 kB)

Obrázek 34: Racionální kvadratické KPCA s elbow technikou 98%

(a) PCA (14,6 kB) (b) $d = 1$, $c = 0$, centrovaná matice (12,9 kB) (c) $d = 1$, $c = 0$ (9,9, kB)

Obrázek 35: Polynomiální KPCA s elbow technikou 98%

Tabulka 14: Porovnání kernel metod

Název kernelu	**Porovnání s PCA**	**Verdikt**
Exponenciální	Srovnatelné při $\sigma = 40$	Použitelný pro kompresi.
Cauchyho	Srovnatelné při $\sigma = 1000$	Použitelný pro kompresi.
Laplaceův	Srovnatelné při $\sigma = 7400$	Použitelný pro kompresi.
Gaussovský	Srovnatelné při $\sigma = 0,000001$	Použitelný pro kompresi.
Mocninový	Rozdílná kvalita obrázků o stejné velikosti.	Nelze použít pro kompresi.
Logaritmický	Rozdílná kvalita a větší velikost výsledného obrázku.	Nelze použít pro kompresi.
Zobecněný T-studentův kernel	U $d \in < 1, 5 >$ se vytváří kvalitní komprese, avšak malá.	Lze použít pro kompresi do určité velikosti.
Kernel založený na hyperbolickém tangentu	Nelze porovnat.	Nelze použít pro kompresi.
Multikvadratický kernel	U $c > 260$ se vytváří kvalitní komprese, avšak malá.	Lze použít pro kompresi do určité velikosti.
Inverzní multikvadratický	Srovnatelné při $c = 600$.	Použitelný při kompresi.
Racionální kvadratický	Srovnatelné při $c = 800000$.	Použitelný při kompresi.
Polynomiální kernel	Rozdílná kvalita.	Nelze použít pro kompresi.

E Porovnání jednotlivých metod

E.1 PCA - Kovarianční a korelační matice

Z obrázku 36 lze vypozorovat, že při použití korelační matice roste velikost a klesá kvalita komprimovaného obrázku. Při použití kovarianční matice je naopak kvalita dobrá a velikost menší než u obrázků, který používá korelační matici.

(a) Kovarianční matice, elbow technika 98% (14,6 kB)

(b) Korelační matice, elbow technika 98% (14,9 kB)

Obrázek 36: Použití kovarianční a korelační matice u metody PCA

E.2 CUR - Optimální procento zachovaných řádků a sloupců

Na obrázku 37 lze vidět výsledky náhodného algoritmu CUR. První obrázek zobrazuje výsledek toho, kdy bylo zvoleno malé procento zachovaných řádků. Objekty nejsou natolik zřetelné jako u obrázku třetího, kde je vše zvoleno optimálně. Obrázek je kvalitní, i když je součet procent blízký hranici 65%. Druhý obrázek naopak demonstruje špatně zvolený součet procent. Lze vypozorovat, že jeden z objektů (pes u řeky) není vůbec zřetelný.

(a) 11% řádků a 81% procent sloupců (14 kB)

(b) 24% řádků a 21% procent sloupců (14 kB)

(c) 34% řádků a 35% procent sloupců (14,6 kB)

Obrázek 37: Náhodný algoritmus CUR

Obrázek 38demonstruje výsledky algoritmu CUR pomocí hrubé síly. Na prvním obrázku lze vidět výsledek toho, kdy byl použit malý součet procent zachovaných řádků a sloupců. Výsledek není moc kvalitní ve srovnání s tím, že tento součet stačil u náhodného algoritmu na kvalitní výsledek. Na druhém obrázku lze v dolní části vypozorovat „kazy“, které mírně znehodnocují obrázek. Poslední obrázek díky vysokému součtu procent demonstruje optimální výsledek.

(a) 35% řádků a 39% procent sloupců (15,9 kB)

(b) 70% řádků a 70% procent sloupců (16,6 kB)

(c) 87% řádků a 85% procent sloupců (16,9 kB)

Obrázek 38: CUR algoritmus hrubé síly

Různé výsledky algoritmu CUR-L2 znázorňuje obrázek 39. První obrázek zobrazuje, stejně jako u předchozí sekvence obrázků, špatně zvolený součet procent. U algoritmu CUR-L2 je neobvyklé to, že vybere určitou část obrázku, která mu přijde důležitá a postupně od ní projasňuje celý obrázek. Tento jev lze vypozorovat z druhého obrázku, kdy algoritmus vybral jako důležitou část obrázku muže s vozem. Poslední obrázek demonstruje již zmíněné vyjasnění kolem důležité části. Na tomto obrázku je již jasně viditelný pes u řeky.

(a) 13% řádků a 36% procent sloupců (14,9 kB)

(b) 29% řádků a 91% procent sloupců (14,9 kB)

(c) 51% řádků a 68% procent sloupců (16,3 kB)

Obrázek 39: CUR - L2 algoritmus

E.3 NMF - Volba optimálního aproximačního parametru

Sekvence obrázků na obrázku 40 demonstruje postupné vylepšení kvality obrázku. U prvního obrázku je aproximační parametr velmi malý, ale již zde lze vidět, jaké objekty se zhruba budou vyskytovat na obrázku, pokud bude kvalitnější. Při $r = 35$ se na obrázku vykresluje i pes u řeky. Poslední obrázek zobrazuje požadovaný obrázek pro experiment 7.3. Všechny objekty

jsou viditelné a při velkém přiblížení lze vidět mezi domem a stromem ženu, která pere v řece prádlo.

(a) $r = 15$ (10,7 kB) (b) $r = 35$ (12,2 kB) (c) $r = 85$ (14,6 kB)

Obrázek 40: Volba parametru r u metody NMF

E.4 NMF - Volba parametru ϵ

Na obrázku 41 lze vidět vliv tolerančního parametru ϵ na kvalitu obrázku. Na prvním obrázku je ϵ poměrně velké, a tak je kvalita obrázku nižší. Jistým způsobem jsou viditelné tři posuzované objekty (dům, vůz a pes) ze čtyř. Viditelná není žena, která pere v řece prádlo. Mezi dalšími dvěma obrázky je rozdíl nepatrný, tři výše zmíněné objekty jsou vyjasněné a pomalu se vyjasňuje i žena s prádlem.

(a) $r = 85$ a $\epsilon = 100000$ (13,1 kB) (b) $r = 85$ a $\epsilon = 1000$ (14,6 kB) (c) NMF - $r = 85$ a $\epsilon = 100$ (14,8 kB)

Obrázek 41: Volba parametru ϵ u metody NMF

E.5 Rychlosti běhu algoritmů vybraných metod

Tabulka 15: Rychlosti běhu algoritmů vybraných metod

Metoda	Čas běhu (s)
SVD	0,6
CUR (Random)	1
CUR (Bruteforce)	1
CUR-L2	1,1
PCA	1,6
Racionální kvadratické KPCA	6,5
Inverzní multikvadratické KPCA	6,5
Cauchyho KPCA	6,6
Laplaceovo KPCA	6,6
Exponenciální KPCA	6,7
Gaussovské KPCA	9,8
NMF	375